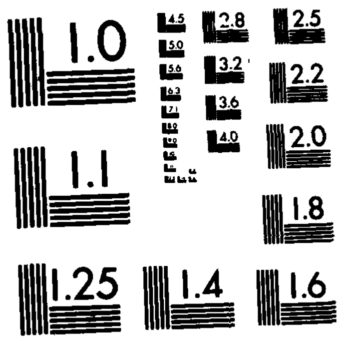
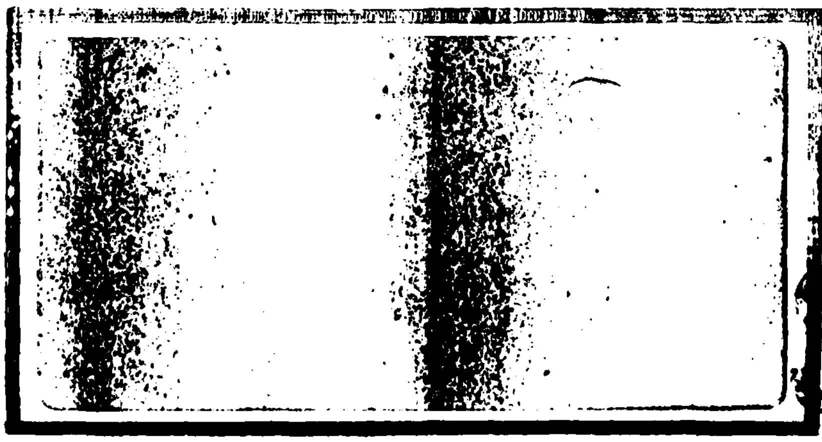
AD-A124 871 HIGH-GAIN ERROR ACTUATED FLIGHT CONTROL SYSTEMS FOR CONTINUOUS LINEAR MUL..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. T LEWIS 1/2

UNCLASSIFIED DEC 82 AFIT/GAE/EE/82D-1 F/G 1/3 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A



This document has been approved for public release and sale; its distribution is unlimited.

"Original contains color plates: All DTIC reproductions will be in black and white"

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

(1)

AFIT/GAE/EE/82D-1

HIGH-GAIN ERROR ACTUATED FLIGHT
CONTROL SYSTEMS FOR CONTINUOUS LINEAR
MULTIVARIABLE PLANTS

THESIS

AFIT/GAE/EE/82D-1

Thomas Lewis
2Lt USAF

AFIT/GAE/EE/82D-1

HIGH-GAIN ERROR ACTUATED FLIGHT CONTROL SYSTEMS FOR CONTINUOUS LINEAR MULTIVARIABLE PLANTS

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

Thomas Lewis, B.S.A+AE.

2Lt USAF

Graduate Aeronautical Engineering

December 1982

Approved for public release; distribution unlimited

Acknowledgements

I would like to thank my thesis advisors, Dr. John D'Azzo and Dr. Constantine Houpis, for all their encouragement and guidance. I would also like to thank the Flight Dynamics Laboratory for sponsoring my thesis, especially the Control Techniques Group and Control Data Group for sharing their facilities with me.

Dedication

I dedicate this thesis to my grandfather William Lewis, a self taught engineer, who was an inspiration throughout my engineering education.

Contents

Illusrations

Figures Page

Tables

Illustrations

Symbols

$\underline{A}$ - State Matrix

$\underline{A}_c$ - Actuator Contant Matrix

$\underline{A}_{CL}$ - Closed-loop State Matrix

$\underline{AA}$ - Generalized Eigenvalue Problem Matrix

b - Wing Span

$\underline{B}$ - Input Matrix

$\underline{B}_{CL}$ - Closed-loop Input Matrix

$\underline{BB}$ - Generalized Eigenvalue Problem Matrix

$\underline{c}_i$ - i^{th} Row of Output Matrix

C - Set of Complex Numbers

$\underline{C}$ - Output Matrix

$\underline{C}_{CL}$ - Closed-loop Output Matrix

C_l - Roll Moment Coeficient

C_n - Yaw Moment Coeficient

C_y - Side Force Coeficient

$\underline{d}$ - Riccati Equation Vector Due to Tracking

dt - Time Derivitive

$\underline{e}$ - Error Vector

$\underline{F}$ - Measurement Output Matrix

g - Acceleration Due to Gravity

- Forward- oop Gain Parameter

$\underline{G}$ - Transfer Function

$\underline{G}_A$ - Transfer Function for Actuator Dynamics

$\underline{H}$ - Weighting Matrix

I_i - Identity Matrix of Rank i

I_{xx} - Moment of Inertia x body axis

I_{xz} - Cross Product of Inertia x and z body axis

I_{zz} - Moment of Inertia z body axis

J - Performance Index

$\underline{K}$ - Controller Matrix

- Feedback Gain Matrix

- Random Matrix

ℓ - Number of Outputs

L - Roll Moment

L' - Roll Moment with Inertia Terms

Symbols

m	- Number of Inputs
$\underline{M}$	- Measurement Matrix
n	- Number of States
N	- Yaw Moment
N'	- Yaw Moment with Inertia Terms
p	- Roll Rate
$\underline{P}$	- Riccati Equation Solution Matrix
Q	- Dynamic Pressure
$\underline{Q}$	- Weighting Matrix
r	- Yaw Rate
R	- Set of Real Numbers
$\underline{R}$	- Weighting Matrix
s	- Laplace Transformation Parameter
S	- Wing Area
t	- Time
t_f	- Final Time
$\underline{u}$	- Input Vector
$\underline{u}_A$	- Input Vector with Actuator Dynamics
$\underline{u}^*$	- Optimal Input Vector
U	- Aircraft Velocity
v	-Pertubation Side Velocity
$\underline{v}$	- Command Vector
w	- Weight
$\underline{w}$	- Measurement Vector
$\underline{x}$	- State Vector
$\underline{y}$	- Output Vector
Y	- Side Force with v as a State
Y*	- Side Force with β as a State
$\underline{z}$	- Time Integral of Error Vector
	- Generalized Eigenvector
z_1	- First Set of Slow Eigenvalues
z_2	- Second Set of Slow Eigenvalues
z_3	- Set of Fast Eigenvalues

Symbols

α_o	- Trim Angle of Attack
β	- Side-Slip Angle
γ	- Flight Path Angle
$\underline{\Gamma}$	- Asymptotic Transfer Function
$\underline{\tilde{\Gamma}}$	- Slow Asymptotic Transfer Function
$\underline{\hat{\Gamma}}$	- Fast Asymptotic Transfer Function
ϕ	- Roll Angle
δ_a	- Aileron Deflection
δ_c	- Cannard Deflection
δ_{dt}	- Differential Tail Deflection
δ_f	- Flaperon Deflection
δ_r	- Rudder Deflection
ε	- Pertubation Parameter
	- " is a member of "
$\underline{\xi}$	- Eigenvector
θ_o	- Trim Flight Path Angle
λ	- Eigenvalue
ρ	- Air Density
σ	- Fast Pole
$\underline{\Sigma}$	- Matrix of Fast Poles
τ	- Time Dummy Argument
$\underline{\omega}$	- Portion of Null Space
ψ	- Pointing Angle
$\dot{\underline{\ }}$	- Time Derivitive of a Vector
N	- Null Space
det	- Determinant of a Matrix
$\lvert\ \rvert$	- Determinant of a Matrix
$[\ ]^T$	- Transpose of a Matrix
$[\ ]^{-1}$	- Inverse of a Matrix

Subscripts

b	- Body Axis
s	- Stability Axis
ss	- Steady-State

Abstract

The theory of high-gain error actuated feedback control, developed by Porter and Bradshaw, is applied to the design of various lateral-directional decoupling flight control systems for an advanced aircraft. The controllers developed in this report utilize output feedback with proportional plus integral control to produce desirable closed-loop responses with minimal interactions between outputs. Because of the structure of the system, measurement variables in addition to the outputs are necessary to apply this method. The report examines controller design robustness by varying the flight conditions or maneuver commands from the ones the controller is specifically designed for, and then judges system performance. The results show that the controller is robust with respect to varying flight conditions, but is not robust with respect to varying maneuver commands. This report also examines the effect of first-order actuator dynamics in the system model. Actuator dynamics are shown to significantly affect the control system response, indicating that a simplified model, without actuators, is not desireable in one's control design scheme. Also a computer program to determine transmission zeros and decoupling zeros is developed.

HIGH-GAIN ERROR ACTUATED FLIGHT
CONTROL SYSTEMS FOR CONTINUOUS LINEAR
MULTIVARIABLE PLANTS

I. Introduction

Advanced Fighter Technology Integration (AFTI) is the culmination of the most recent technology advances in flight control and avionics. The AFTI-16 aircraft, which incorporates this technology, is presently undergoing flight tests. A new digital flight control system, along with additional control surfaces, such as vertical cannards, allow the aircraft to perform direct force, decoupled maneuvers.

This report contains an investigation of a new controller design method for the AFTI-16. Specifically, an analogue controller is designed to perform four lateral-directional decoupled maneuvers. These maneuvers are horizontal translation, wings level turn, constant altitude coordinated turn, and yaw pointing. The maneuvers can be described by three output variables: side-slip angle, β; roll angle, ϕ; and yaw rate, r. Initially all output variables are zero. For horizontal translation the side-slip angle, β, changes, while ϕ and r remain unchanged. In a wings level turn the yaw rate, r, changes, while β and ϕ remain unchanged. In a constant altitude coordinated turn the yaw rate, r, is proportional to the roll angle, ϕ,

$$r = \frac{g}{U} \sin \phi, \tag{1-1}$$

while β remains zero. For yaw pointing the time rate of change of the side-slip angle, β, must be equal in magnitude and opposite in sign to

the yaw rate, r, while ϕ remains zero.

Background

Controllers have been designed for single input/single output (SISO) systems using conventional techniques for a long time. The advance of computational technology readily permits the design of controllers for multiple input/multiple output (MIMO) systems. A MIMO system design offers a greater amount of control and greater precision. Conventional control employs widely accepted design techniques, such as root-locus and frequency response analysis. Multivariable control design is still growing and a number of different theories and design methods have been developed. In this report the design method developed and proposed by Dr Brian Porter of the University of Salford, England is utilized and evaluated.

Other multivariable control methods presently available utilize optimal control, entire eigenstructure assignment, and frequency domain techniques.

In the optimal control design method (Ref.2) a control law is found by minimizing the value of the performance index or cost functional

$$J = 1/2 \int_0^{t_f} \Big[[\underline{y}(t) - \underline{v}(t)]^T \underline{Q}(t) [\underline{y}(t) - \underline{v}(t)] + \underline{u}^T(t)\ \underline{R}(t)\ \underline{u}(t) \Big] dt + 1/2 [\underline{y}(t_f) - \underline{v}(t_f)]^T\ \underline{H}\ [\underline{y}(t_f) - \underline{v}(t_f)] \tag{1-2}$$

for the system described by the state and output equations

$$\dot{\underline{x}}(t) = \underline{A}\ \underline{x}(t) + \underline{B}\ \underline{u}(t)$$

$$\underline{y}(t) = \underline{C}\ x(t) \tag{1-3}$$

where $\underline{y}(t)$ is the output vector.

$\underline{v}(t)$ is the command vector, and $\underline{Q}(t)$, $\underline{R}(t)$, and $\underline{H}$ are symmetric positive definite quadratic weighting matrices. The optimal control law that minimizes J is

$$\underline{u}^*(t) = \underline{K}(t)\ \underline{x}(t) - \underline{R}^{-1}(t)\ \underline{B}^T\ \underline{d}(t) \tag{1-4}$$

The time varying feedback gain matrix is

$$\underline{K}(t) = -\ \underline{R}^{-1}(t)\ \underline{B}^T\ \underline{P}(t) \tag{1-5}$$

where $\underline{P}(t)$ is the solution of a nonlinear time varying matrix differential equation, commonly known as the Riccati Equation:

$$\begin{aligned}\dot{\underline{P}}(t) = -\underline{A}^T\ P(t) - \underline{P}(t)A + \underline{P}(t)\underline{B}\ \underline{R}^{-1}(t)\ \underline{B}^T\underline{P}(t) \\ -\underline{C}^T\ \underline{Q}(t)\ \underline{C}\end{aligned} \tag{1-6}$$

where $P(t_f) = \underline{C}^T\ \underline{H}\ \underline{C}$.

The vector $\underline{d}(t)$ is the solution to the matrix differential equation

$$\begin{aligned}\dot{\underline{d}}(t) = \underline{A}^T\underline{d}(t) + \underline{K}(t)\ \underline{B}\ \underline{R}^{-1}(t)\underline{B}^T\underline{d}(t) \\ +\ \underline{C}^T\ \underline{Q}(t)\ \underline{C}\end{aligned} \tag{1-7}$$

where $\underline{d}(t_f) = -\underline{C}^T\ \underline{H}\ \underline{C}\ \underline{v}(t_f)$.

Due to the boundary conditions on the two matrix differential equations, they must be integrated backwards in time. This requires that command input information be known in the future. The computation required for the numerical solutions to these equations are extremely costly

and difficult to achieve. Also, selection of the positive definite weighting matrices, $\underline{Q}(t)$, $\underline{R}(t)$, and $\underline{H}$, must be made. This is usually accomplished by a trial and error iterative process, requiring repeated integrations of the matrix differential equations.

In the entire eigenstructure assignment design method (Ref 11), the designer selects the system closed-loop eigenvalues, λ_i, and corresponding eigenvectors, $\underline{\xi}_i$. Given the system described by eq. 1.3, a state feedback control law

$$\underline{u}(t) = \underline{K}\,\underline{x}(t) + \underline{v}(t) \tag{1-8}$$

is assumed. Substituting this control law into the state equations yields the closed-loop system

$$\underline{\dot{x}}(t) = \left[\underline{A} + \underline{B}\,\underline{K}\right]\underline{x}(t) + \underline{B}\,\underline{v}(t) \tag{1-9}$$

The feedback gain matrix, $\underline{K}$, is

$$\underline{K} = \left[\underline{\omega}_1, \underline{\omega}_2, \ldots, \underline{\omega}_n\right]\left[\underline{\xi}_1, \underline{\xi}_2, \ldots, \underline{\xi}_n\right]^{-1} \tag{1-10}$$

The feedback gain matrix of eq. 1-10 produces a closed-loop system with the selected eigenvalues and corresponding eigenvectors, provided all eigenvectors, $\underline{\xi}_i$, and vectors, $\underline{\omega}_i$, satisfy the relation

$$\begin{bmatrix} \underline{\xi}_i \\ \\ \underline{\omega}_i \end{bmatrix} \varepsilon\, N\left[\underline{A} - \lambda_i I_n\,,\ \underline{B}\right] \tag{1-11}$$

Extreme care must be taken when selecting the eigenvectors from the null space of $\left[\underline{A} - \lambda_i\, I,\ B\right]$, in order that the matrix $\left[\underline{\xi}_1, \underline{\xi}_2, \ldots \underline{\xi}_n\right]^{-1}$

exists. Also, the eigenvalues and eigenvectors must be chosen to minimize the amount of interaction between outputs. Selection of the eigenvalues, λ_i, and eigenvectors, $\underline{\xi}_i$, is an art, requiring insight and a lot of trial and error iterations.

Porter's design method, described in Chapter II, often produces a successful controller design with less trial and error iterations than the two methods described above.

Problem Statement and Approach

It is desired to design and evaluate a flight controller, using Porter's design method (Ref 7), for an AFTI-16 aircraft which simulates maneuvers of horizontal translation, wings level turn, constant altitude coordinated turn, and yaw pointing. The evaluation consists of examining robustness and effect of first-order actuators in the simulation.

A robust controller design can perform satisfactorily at conditions other than the one it is specifically designed for. In this report robustness is examined by using a controller designed for a yaw pointing maneuver at a specific design flight condition, and using it without change in the simulation of a yaw pointing maneuver at other flight conditions. Also the same controller is used in the simulation of the three other maneuvers at the design flight condition. The performance of the three maneuvers is compared with the performance of controllers designed to perform the specific maneuver.

Three different controllers are designed to perform the yaw pointing maneuver. One at a low dynamic pressure, Q, design flight condition, another at a medium dynamic pressure design flight condition, and the

third at a high dynamic pressure design flight condition. Each of these controllers are then included in the simulation of a yaw pointing maneuver at nine other flight conditions, covering a spectrum of dynamic pressures.

A successful design is based on several criteria. In this report the following specifications are established: First, the closed-loop system must be stable. Second, all transient responses should be within 98% of their steady-state value in three seconds. Third, the steady-state response should achieve the commanded value with no observable error. Fourth, the control surfaces are not commanded beyond a pre-set limit.

Assumptions

The following assumptions are made to simplify the computations and analysis:

1. The earth is flat and non-rotating.
2. The atmosphere is at rest with respect to the earth.
3. The acceleration due to gravity is constant.
4. The aircraft is a rigid body.
5. The aircraft has a constant mass.
6. The aircraft performs maneuvers while flying straight with wings level, unless otherwise specified.
7. The motion of the aircraft about a straight and level trim condition can be adequately modeled by a linear dynamical system of equations.
8. The dynamics of the actuators can be adequately modeled by a first-order differential equation.
9. The side slip angle, β, can be measured.

Organization

First, the theory used in Porter's design is described, and the method for developing a controller is discussed. Also included in Chapter II are methods for including first-order actuator dynamics in the model. Chapter III presents the results of applying this method to an aircraft model. The effects of varying flight conditions and maneuver commands are described in detail. Also the effects of actuator dynamics in the simulation are discussed. Chapter IV discusses the results and provides recommendations for further study.

Included in the Appendices are discussion of the aircraft equations of motion, a computer program to calculate system zeros, and the computer program used to perform design iterations and simulations.

II. Theoretical Development

Porter's design methods (Ref 7) are both conceptually and computationally simple. These techniques utilize singular perturbation methods, and are applicable to the design of both analogue and digital controllers. As stated in the introduction these methods are applied to tracking systems incorporating high-gain error actuated analogue controllers.

High-gain frequently has the advantage of making the closed-loop system robust, or insensitive to plant parameter variations. Also, high-gain can produce a tight tracking loop which may be desirable from the viewpoint of disturbance rejection. Robustness is examined in this report, but disturbance rejection is not.

Error actuated control minimizes the difference between the commands and outputs. The closed-loop behavior becomes increasingly tight and non-interacting as the gain parameter is increased.

Singular perturbation methods are used to exhibit the asymptotic structure of the transfer function matrices. These methods examine the overall effect of letting certain parameters become smaller and smaller. By letting the perturbation parameter, ε, be the reciprocal of the forward path gain, g, the results of singular perturbation theory can be applied, since as $\varepsilon \rightarrow 0$, $g = 1/\varepsilon \rightarrow \infty$. With singular perturbation theory the asymptotic properties of the closed-loop transfer functions can be examined to determine the location of the closed-loop poles, as the gain becomes increasingly large. These asymptotic closed-loop poles determine the system response.

The tracking system consists of a linear multivariable plant governed on the continuous time set $\mathcal{T} = (0,\infty)$ represented by state and output equations of the form

$$\dot{\underline{x}}(t) = \underline{A}\,\underline{x}(t) + \underline{B}\,\underline{u}(t)$$
$$\underline{y}(t) = \underline{C}\,\underline{x}(t) \tag{2.1}$$

The state and output equations, eq. 2.1, expressed in expanded form are

$$\begin{bmatrix} \dot{\underline{x}}_1(t) \\ \dot{\underline{x}}_2(t) \end{bmatrix} = \begin{bmatrix} \underline{A}_{11} & \underline{A}_{12} \\ \underline{A}_{21} & \underline{A}_{22} \end{bmatrix} \begin{bmatrix} \underline{x}_1(t) \\ \underline{x}_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \underline{B}_2 \end{bmatrix} \underline{u}(t) \tag{2.2}$$

$$\underline{y}(t) = \begin{bmatrix} \underline{C}_1 & \underline{C}_2 \end{bmatrix} \begin{bmatrix} \underline{x}_1(t) \\ \underline{x}_2(t) \end{bmatrix} \tag{2.3}$$

Where $\underline{B}_2$ is a square non-singular matrix. Any system of the form of eq. 2.1 can, by proper state transformation, be put into the form of eqs. 2.2 and 2.3. In an improper system, where $\underline{C}_2\underline{B}_2$ is rank deficient, the output measurements which track the command inputs are

$$\underline{w}(t) = \begin{bmatrix} \underline{C}_1 + \underline{M}\,\underline{A}_{11}, & \underline{C}_2 + \underline{M}\,\underline{A}_{12} \end{bmatrix} \begin{bmatrix} \underline{x}_1\ (t) \\ \underline{x}_2\ (t) \end{bmatrix} \tag{2.4}$$

$$= \begin{bmatrix} \underline{F}_1, & F_2 \end{bmatrix} \begin{bmatrix} \underline{x}_1\ (t) \\ \underline{x}_2\ (t) \end{bmatrix} \tag{2.5}$$

where $\underline{M}$ is an extra plant measurement matrix which must be designed so that $\underline{F}_2\ \underline{B}_2$ has full rank. The trackers examined in this report are

improper, requiring a measurement matrix, $\underline{M}$. The state vector $\underline{x}(t) = (\underline{x_1}\ (t),\ \underline{x_2}\ (t))^T$ in these equations represent the kinematic variables used in the aircraft equations of motion

$$\underline{x_1}\ (t) = \phi(t) \tag{2.6}$$

$$\underline{x_2}\ (t) = (\beta(t),\ p(t),\ r(t))^T \tag{2.7}$$

The high-gain error actuated analogue controller is governed on $\mathcal{T}$ by a proportional plus integral control law equation of the form

$$\underline{u}\ (t) = g(\underline{K_o}\underline{e}(t) + \underline{K_1}\underline{z}(t)) \tag{2.8}$$

where $\underline{K_o}$ and $\underline{K_1}$ are controller gain matrices, $\underline{e}\ (t)$ is the error vector

$$\underline{e}\ (t) = \underline{v}\ (t) - \underline{w}\ (t) \tag{2.9}$$

$\underline{v}\ (t)$ is the command vector, and $\underline{z}\ (t)$ is the time integral of $\underline{e}\ (t)$.

$$\dot{\underline{z}}\ (t) = \underline{e}\ (t) = \left[\underline{v}\ (t) - \underline{w}\ (t)\right] = \left[\underline{v}\ (t) - \underline{F_1}\underline{x_1}\ (t) - \underline{F_2}\underline{x_2}(t)\right] \tag{2.10}$$

A functional block diagram of the complete tracker is shown in Figure 2.1.

It is desired that the control input vector, $\underline{u}(t)$, cause the output vector, $\underline{y}(t)$, to track any constant command input vector, $\underline{v}(t)$, so that

$$\lim_{t \to \infty} (\underline{v}\ (t) - \underline{y}\ (t)) = 0 \tag{2.11}$$

Such tracking is possible using the controller form of eq. 2.8, due to the fact that the error vector, $\underline{e}\ (t)$, assumes the steady-state value (Ref 7).

$$\lim_{t \to \infty} \underline{e}(t) = \lim_{t \to \infty} (\underline{v}(t) - \underline{w}(t)) = 0 \tag{2.12}$$

The steady-state value of $\underline{w}(t)$ is

$$\lim_{t \to \infty} \underline{w}(t) = \lim_{t \to \infty} (\underline{F_1}\, \underline{x_1}(t) + \underline{F_2}\, \underline{x_2}(t)) \tag{2.13}$$

$$= \lim_{t \to \infty} ((\underline{C_1} + \underline{M}\, \underline{A_{11}})\underline{x_1}(t) + (\underline{C_2} + \underline{M}\, \underline{A_{12}})\underline{x_2}(t)) \tag{2.14}$$

$$= \lim_{t \to \infty} ((\underline{C_1}\, \underline{x_1}(t) + \underline{C_2}\, \underline{x_2}(t)) + \underline{M}(\underline{A_{11}}\underline{x_1}(t) + \underline{A_{12}}\underline{x_2}(t))) \tag{2.15}$$

$$= \lim_{t \to \infty} ((\underline{C_1}\, \underline{x_1}(t) + \underline{C_2}\, \underline{x_2}(t)) + \underline{M}\, \underline{\dot{x}_1}(t)) \tag{2.16}$$

For a stable system with a constant vector input, the states reach constant values, and therefore it is evident from the state equation, eq. 2.2, that

$$\lim_{t \to \infty} \underline{\dot{x}_1}(t) = \lim_{t \to \infty} (\underline{A_{11}}\underline{x_1}(t) + \underline{A_{12}}\underline{x_2}(t)) = 0 \tag{2.17}$$

Therefore eq. 2.16 yields

$$\lim_{t \to \infty} \underline{w}(t) = \lim_{t \to \infty} (\underline{C_1}\, \underline{x_1}(t) + \underline{C_2}\, x_2(t)) \tag{2.18}$$

$$= \lim_{t \to \infty} (\underline{y}(t)) \tag{2.19}$$

Thus, in the steady-state the output, $\underline{y}(t)$ tracks the command input, $\underline{v}(t)$, for arbitrary initial conditions. Equations 2.12 through 2.19 show that tracking is possible for any measurement matrix, $\underline{M}$, provided that the condition that $\underline{F}_2\ \underline{B}_2$ have full rank is satisfied.

In the above equations, $\underline{x}_1(t) \in R^{n-\ell}$, $\underline{x}_2(t) \in R^{\ell}$, $\underline{u}(t) \in R^{\ell}$, $\underline{y}(t) \in R^{\ell}$, $\underline{w}(t) \in R^{\ell}$, $\underline{e}(t) \in R^{\ell}$, $\underline{z}(t) \in R^{\ell}$, $\underline{v}(t) \in R^{\ell}$, $\underline{A}_{11} \in R^{(n-\ell)x(n-\ell)}$ $\underline{A}_{21} \in R^{\ell x(n-\ell)}$, $\underline{A}_{22} \in R^{\ell x \ell}$, $\underline{B}_2 \in R^{\ell x \ell}$, $\underline{C}_1 \in R^{\ell x(n-\ell)}$, $\underline{C}_2 \in R^{\ell x \ell}$, $\underline{F}_1 \in R^{\ell x(n-\ell)}$, $\underline{F}_2 \in R^{\ell x \ell}$, $\underline{K}_o \in R^{\ell x \ell}$, $\underline{K}_1 \in R^{\ell x \ell}$, $\underline{M} \in R^{\ell x(n-\ell)}$.

The closed-loop equations are formed by combining eqs. 2.2, 2.3, 2.5, 2.8, and 2.10, to yield

$$\begin{bmatrix} \dot{\underline{z}}(t) \\ \dot{\underline{x}}_1(t) \\ \dot{\underline{x}}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & -\underline{F}_1 & -\underline{F}_2 \\ 0 & \underline{A}_{11} & \underline{A}_{12} \\ g\underline{B}_2\ \underline{K}_1, & \underline{A}_{21} - gB_2\ K_o\ F_1, & \underline{A}_{22} - g\underline{B}_2\ \underline{K}_o\ \underline{F}_2 \end{bmatrix} \begin{bmatrix} \underline{z}(t) \\ \underline{x}_1(t) \\ \underline{x}_2(t) \end{bmatrix}$$

$$+ \begin{bmatrix} I_\ell \\ 0 \\ g\underline{B}_2\ \underline{K}_o \end{bmatrix} \underline{v}(t) \tag{2.20}$$

$$\underline{y}(t) = \left[(0,\ \underline{C}_1,\ \underline{C}_2)\right] \begin{bmatrix} \underline{z}(t) \\ \underline{x}_1(t) \\ \underline{x}_2(t) \end{bmatrix} \tag{2.21}$$

The closed-loop transfer function matrix relating the plant output vector to the command input vector of the system governed by eqs 2.20 and 2.21 is

$$\underline{G}(\lambda) = (0,\ C_1,\ C_2)\begin{bmatrix} \lambda I_\ell & , & \underline{F}_1 & , & \underline{F}_2 \\ 0 & , & \lambda I_{n-\ell}-\underline{A}_{11} & , & -\underline{A}_{12} \\ -g\underline{B}_2\ \underline{K}_1, & & -\underline{A}_{21}+g\underline{B}_2\underline{K}_o\ F_1, & & -\lambda I_\ell-\underline{A}_{22}+gB_2K_oF_2 \end{bmatrix}^{-1} \begin{bmatrix} I_\ell \\ 0 \\ g\underline{B}_2K_o \end{bmatrix} \tag{2.22}$$

which is equivalent to

$$\underline{G}(\lambda) = \underline{C}_{CL}\left[\lambda I_{n+\ell} - \underline{A}_{CL}\right]^{-1} \underline{B}_{CL} \tag{2.23}$$

where

$$\underline{A}_{CL} = \begin{bmatrix} \underline{A}_1 & \underline{A}_2 \\ \underline{A}_3/\varepsilon & \underline{A}_4/\varepsilon \end{bmatrix} \tag{2.24}$$

$$\underline{B}_{CL} = \begin{bmatrix} \underline{B}_1 \\ \underline{B}_2/\varepsilon \end{bmatrix} \tag{2.25}$$

$$\underline{C}_{CL} = \left[\ \underline{C}_1,\ \underline{C}_2\right] \tag{2.26}$$

and

$$\underline{A}_1 = \begin{bmatrix} 0 & -\underline{F}_1 \\ 0 & \underline{A}_{11} \end{bmatrix} \tag{2.27}$$

$$\underline{A}_2 = \begin{bmatrix} -\underline{F}_2 \\ \underline{A}_{12} \end{bmatrix} \tag{2.28}$$

$$\underline{A}_3 = \left[\ +\underline{B}_2\ \underline{K}_1,\ \underline{A}_{21}\varepsilon - \underline{B}_2\ \underline{K}_o\ \underline{F}_1\right] \tag{2.29}$$

$$\underline{A}_4 = \left[\ \underline{A}_{22}\varepsilon - \underline{B}_2\ \underline{K}_o\ \underline{F}_2\right] \tag{2.30}$$

$$\underline{B}_1 = \begin{bmatrix} I_\ell \\ 0 \end{bmatrix} \tag{2.31}$$

$$\underline{B}_2 = \left[\underline{B}_2\ \underline{K}_o\right] \tag{2.32}$$

$$\underline{C}_1 = \left[0,\ \underline{C}_1'\right] \tag{2.33}$$

$$\underline{C}_2 = \left[\underline{C}_2\right] \tag{2.34}$$

By letting the perturbation parameter $\varepsilon = 1/g$ go to zero the transfer function matrix $\underline{G}(\lambda)$, approaches (Ref 7) the asymptotic form

$$\underline{\Gamma}(\lambda) = \tilde{\underline{\Gamma}}(\lambda) + \hat{\underline{\Gamma}}(\lambda) \tag{2.35}$$

where

$$\tilde{\underline{\Gamma}}(\lambda) = \underline{C}_o \left[\lambda I_n - \underline{A}_o\right]^{-1} \underline{B}_o \tag{2.36}$$

and

$$\hat{\underline{\Gamma}}(\lambda) = \underline{C}_2 \left[\lambda I_\ell - g\ \underline{A}_4\right]^{-1} g\ \underline{B}_2 \tag{2.37}$$

The matrices in eq. 2.36 are

$$\underline{A}_o = \left[\underline{A}_1 - \underline{A}_2\ \underline{A}_4^{-1}\underline{A}_3\right] = \begin{bmatrix} -\ \underline{K}_o^{-1}\ \underline{K}_1 & 0 \\ \underline{A}_{12}\underline{F}_2^{-1}\underline{K}_o\ \underline{K}_1 & \underline{A}_{11} - \underline{A}_{12}\ \underline{F}_2^{-1}\underline{F}_1 \end{bmatrix} \tag{2.38}$$

$$\underline{B}_o = \left[\underline{B}_1 - \underline{A}_2\ \underline{A}_4^{-1}\ \underline{A}_3\right] = \begin{bmatrix} 0 \\ \underline{A}_{12}\underline{F}_2^{-1} \end{bmatrix} \tag{2.39}$$

$$\underline{C}_o = \left[\underline{C}_1 - \underline{C}_2\ \underline{A}_4^{-1}\ \underline{A}_3\right] = \left[\underline{C}_2\ \underline{F}_2^{-1}\ \underline{K}_o^{-1}\ \underline{K}_1,\ \underline{C}_1 - \underline{C}_2\ \underline{F}_2^{-1}\ \underline{F}_1\right] \tag{2.40}$$

The "slow" modes of the tracking system are defined as those in which the eigenvalues are not direct functions of g. These are the poles of $\tilde{\underline{\Gamma}}(\lambda)$ and are easily found from

$$\det\left[\lambda I_n - \underline{A}_o\right] = 0 \tag{2.41}$$

Since $\underline{A}_o$ is lower block triangular, eq. 2.38 yields two sets of slow poles described by

$$z_1 = \left\{ \lambda \varepsilon C: \left| \lambda \underline{K}_o + \underline{K}_1 \right| = 0 \right\} \tag{2.42}$$

and

$$z_2 = \left\{ \lambda \varepsilon C: \left| \lambda I_{n-\ell} - \underline{A}_{11} + \underline{A}_{12} \underline{F}_2^{-1} \underline{F}_1 \right| = 0 \right\} \tag{2.43}$$

Set z_2 contains the transmission zeros of the system.

The "fast" modes of the tracking system are those in which the eigenvalues are directly dependent on g. These eigenvalues are the poles of $\hat{\underline{\Gamma}}(\lambda)$ and are easily found from

$$\det \left[\lambda I_\ell - g \underline{A}_4 \right] = 0 \tag{2.44}$$

This equation yields the set of fast poles described by

$$z_3 = \left\{ \lambda \varepsilon C: \left| \lambda I_\ell + g \underline{F}_2 \underline{B}_2 \underline{K}_o \right| = 0 \right\} \tag{2.45}$$

In order for tracking to occur, the closed-loop system must obviously be stable, requiring that all the eigenvalues in the sets z_1, z_2 and z_3 be in the left half of the complex plane. Therefore, for sufficiently large gains, the controller matrices, $\underline{K}_o$ and $\underline{K}_1$, and measurement matrix, $\underline{M}$, must be chosen so that $z_1 \varepsilon C^-$, $z_2 \varepsilon C^-$, and $z_3 \varepsilon C^-$.

To achieve a non-interacting system it is desired to have a diagonal asymptotic transfer function matrix $\underline{\Gamma}(\lambda)$. In order for the fast asymptotic transfer function, $\hat{\underline{\Gamma}}(\lambda)$, to be diagonal it is neces-

$$\underline{F}_2 \underline{B}_2 \underline{K}_o = (\underline{C}_2 + \underline{M} \underline{A}_{12}) \underline{B}_2 \underline{K}_o = \underline{\Sigma} = \text{diag } \sigma_1, \sigma_2, \ldots, \sigma_\ell \tag{2.46}$$

where $\sigma_j \varepsilon R^+$ ($j = 1, 2, \ldots, \ell$). By selecting appropriate values for σ_j, the elements of $\underline{K}_o$ can then be determined

$$\underline{K}_o = (\underline{F}_2\ \underline{B}_2)^{-1}\ \underline{\Sigma} \tag{2.47}$$

Thus the fast asymptotic eigenvalues are

$$z_3 = \left\{-\sigma_1\ g,\ -\sigma_2\ g,\ \ldots,\ -\sigma_\ell g\right\} \tag{2.48}$$

$\underline{K}_1$ may be chosen as a positive multiple of $\underline{K}_o$

$$\underline{K}_1 = b\underline{K}_o \tag{2.49}$$

Solving eq. 2.40, the first set of slow asymptotic eigenvalues are

$$z_1 = \left\{-b\right\} \tag{2.50}$$

There are two methods, presented by Ridgley, Banda., and D'Azzo (Ref 9), for selecting a measurement matrix, $\underline{M}$, which diagonalizes the slow asymptotic transfer function, $\underline{\tilde{\Gamma}}(\lambda)$. The first method is used when $n > 2\ell$ and when

$$\det \underline{B}^* = \det \begin{bmatrix} \underline{c}_1^T & \underline{A}'^{d_1}\ \underline{B}' \\ & \vdots \\ \underline{c}_\ell^T & A'^{d_\ell}\ \underline{B}' \end{bmatrix} \neq 0 \tag{2.51}$$

where $\underline{A}' = \underline{A}_{11}$, $\underline{B}' = \underline{A}_{12}$. $\underline{C}' = \underline{C}_1$, $\underline{c}_i^T$ is the ith row of $\underline{C}'$, and $d_i = \min\ (j\text{: } \underline{c}_i^T\ \underline{A}^{1j}\ \underline{B}' \neq 0,\ j = 1, 2, \ldots, n-\ell-1)$ or $d_i = n-\ell-1$ if $\underline{c}_i^T\ \underline{A}^{1j}\ \underline{B}' = 0$ for all j.

The measurement matrix, $\underline{M}$, is chosen to satisfy

$$\underline{F}_2 = (\underline{C}_2 + \underline{M}\ \underline{A}_{12}) = \underline{B}^* \tag{2.52}$$

and

$$\underline{F}_1 = (\underline{C}_1 + \underline{M}\,\underline{A}_{11}) = -\left[\sum_{K=0}^{\delta} \underline{N}_K\, \underline{C}'\, \underline{A}'^{K} - A^*\right] \tag{2.53}$$

where δ is the maximum value of d_i, $\underline{M}_K$ are suitably chosen diagonal matrices, and

$$\underline{A}^* = \begin{bmatrix} \bar{c}_1^{\,T} & \underline{A}'^{\,d_i} \\ & \vdots \\ \underline{C}_\ell^{\,T} & \underline{A}'^{\,d_\ell} \end{bmatrix} \quad \underline{A}' \tag{2.54}$$

It is not always possible to find measurement matrix, $\underline{M}$, which satisfies both eqs. 2.52 and 2.53. In this case it is not possible to diagonalize the slow asymptotic transfer function matrix, $\underline{\tilde{\Gamma}}(\lambda)$.

The second method is used when $n < 2\ell$ and det $\underline{B}^* = 0$. The assignable elements of $\underline{F}_2 = (\underline{C}_2 + \underline{M}\,\underline{A}_{12})$, those containing elements of $\underline{M}$, are permitted non-zero values only if $\underline{B}^*$ has a non-zero element in the corresponding position. Also $\underline{C}_2\, \underline{F}_2^{-1}$ should be chosen to be diagonal. Again it is not always possible to find a matrix $\underline{M}$ which diagonalizes the slow asymptotic transfer function, $\underline{\tilde{\Gamma}}(\lambda)$.

When there is no measurement matrix, $\underline{M}$, which will decouple the system outputs and thus make the asymptotic transfer function, $\underline{\Gamma}(\lambda)$, diagonal, then $\underline{M}$ must be iteratively selected, aiming at a closed-loop system that is stable and nearly non-interactive. This is a limitation of these two methods.

When it is desired to include first-order actuators into the model, two methods are available. One is to add an actuator transfer function to the forward loop path, as shown in Figure 2.2. The other is to augment the open-loop state equations.

The actuator transfer function, in the Laplace domain, for first-order actuators is

$$\underline{G}_A(s) = \underline{A}_c \left[sI + \underline{A}_c \right]^{-1} \tag{2.55}$$

where $\underline{A}_c$ is a diagonal matrix containing the ℓ values of positive actuator constants. The resulting vector differential equation is

$$\dot{\underline{u}}_A(t) = -\underline{A}_c\, \underline{u}_A(t) + \underline{A}_c\, \underline{u}(t) \tag{2.56}$$

Adding the actuator transfer function, $\underline{G}_A$, to the forward path augments the closed-loop equations, eqs. 2.20 and 2.21. Combining eqs. 2.5, 2.8, 2.9, and 2.56 yields

$$\dot{\underline{u}}_A(t) = -\underline{A}_c\, \underline{u}_A(t) + \underline{A}_c \left[g(\underline{K}_o\, \underline{e}(t) + \underline{K}_1\, \underline{z}(t)) \right] \tag{2.57}$$

$$= -\underline{A}_c\, \underline{u}_A(t) + g\underline{A}_c \left[\underline{K}_o (\underline{u}(t) - \underline{F}_1 \underline{x}_1(t) - \underline{F}_2 \underline{x}_2(t) + \underline{K}_1 \underline{z}(t) \right] \tag{2.58}$$

The closed-loop state vector is augmented to include $\underline{u}_A$ as a state. The resulting closed-loop equations are

$$\begin{bmatrix} \dot{\underline{z}}(t) \\ \dot{\underline{x}}_1(t) \\ \dot{\underline{x}}_2(t) \\ \underline{u}_A(t) \end{bmatrix} = \begin{bmatrix} 0, & -\underline{F}_1, & -\underline{F}_2, & 0 \\ 0, & \underline{A}_{11}, & \underline{A}_{12}, & 0 \\ 0, & \underline{A}_{21}, & \underline{A}_{22}, & \underline{B}_2 \\ g\underline{A}_c\underline{K}_1, & -g\underline{A}_c\underline{K}_o\underline{F}_1, & -g\underline{A}_c\underline{K}_o\underline{F}_2, & -A_c \end{bmatrix} \begin{bmatrix} \underline{z}(t) \\ \underline{x}_1(t) \\ \underline{x}_2(t) \\ \underline{u}_A(t) \end{bmatrix} + \begin{bmatrix} I_\ell \\ 0 \\ 0 \\ g\underline{A}_c\underline{K}_o \end{bmatrix} \underline{u}(t) \tag{2.59}$$

$$\underline{y}(t) = \left[0,\ \underline{c}_1,\ \underline{c}_2,\ 0 \right] \begin{bmatrix} \underline{z}(t) \\ \underline{x}_1(t) \\ \underline{x}_2(t) \\ \underline{u}_A(t) \end{bmatrix} \tag{2.60}$$

To add the first-order actuators by augmenting the open-loop state equations, the open-loop state vector is expanded to include $\underline{u}_A$. The resulting open-loop equations are

$$\begin{bmatrix} \dot{\underline{x}}_1'(t) \\ \dot{\underline{x}}_2'(t) \end{bmatrix} = \begin{bmatrix} \underline{A}_{11}' & \underline{A}_{12}' \\ \underline{A}_{21}' & \underline{A}_{22}' \end{bmatrix} \begin{bmatrix} \underline{x}_1'(t) \\ \underline{x}_2'(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \underline{B}_2' \end{bmatrix} \bar{u}(t) \tag{2.61}$$

$$\underline{y}(t) = \begin{bmatrix} \underline{C}_1' & \underline{C}_2' \end{bmatrix} \begin{bmatrix} \underline{x}_1(t) \\ \underline{x}_2(t) \end{bmatrix} \tag{2.62}$$

where

$$\underline{x}_1' = \begin{bmatrix} \underline{x}_1(t) \\ \underline{x}_2(t) \end{bmatrix} \tag{2.63}$$

$$\underline{x}_2' = \begin{bmatrix} \underline{u}_A(t) \end{bmatrix} \tag{2.64}$$

$$\underline{A}_{11}' = \begin{bmatrix} \underline{A}_{11} & \underline{A}_{12} \\ \underline{A}_{21} & \underline{A}_{22} \end{bmatrix} \tag{2.65}$$

$$\underline{A}_{12}' = \begin{bmatrix} 0 \\ \underline{B}_2 \end{bmatrix} \tag{2.66}$$

$$\underline{A}_{21}' = \begin{bmatrix} \underline{0} \end{bmatrix} \tag{2.67}$$

$$\underline{A}_{22}' = \begin{bmatrix} -\underline{A}_c \end{bmatrix} \tag{2.68}$$

$$\underline{B}_2' = \begin{bmatrix} \underline{A}_c \end{bmatrix} \tag{2.69}$$

$$\underline{C}_1' = \begin{bmatrix} \underline{C}_1, \underline{C}_2 \end{bmatrix} \tag{2.70}$$

$$\underline{C}_2' = \begin{bmatrix} \underline{0} \end{bmatrix} \tag{2.71}$$

Equations 2.62 and 2.63 are substituted for eqs. 2.2 and 2.3, then the theory outlined in this chapter is applied.

In this report the first-order actuators are added by augmenting the closed-loop equations.

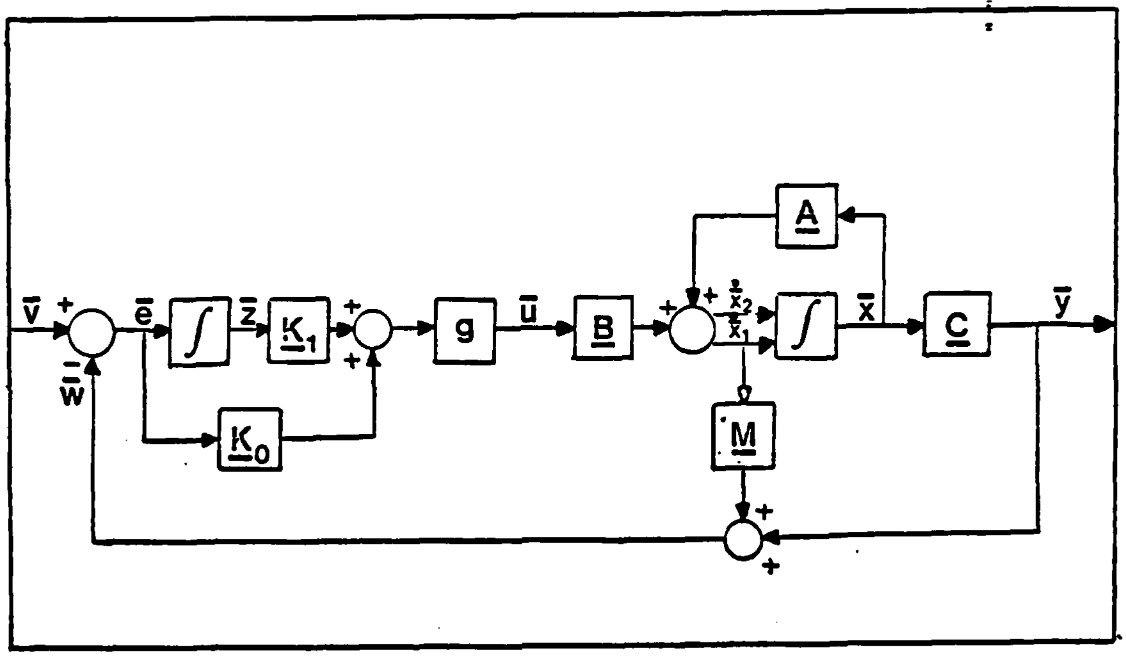


Figure 2-1 Tracking System

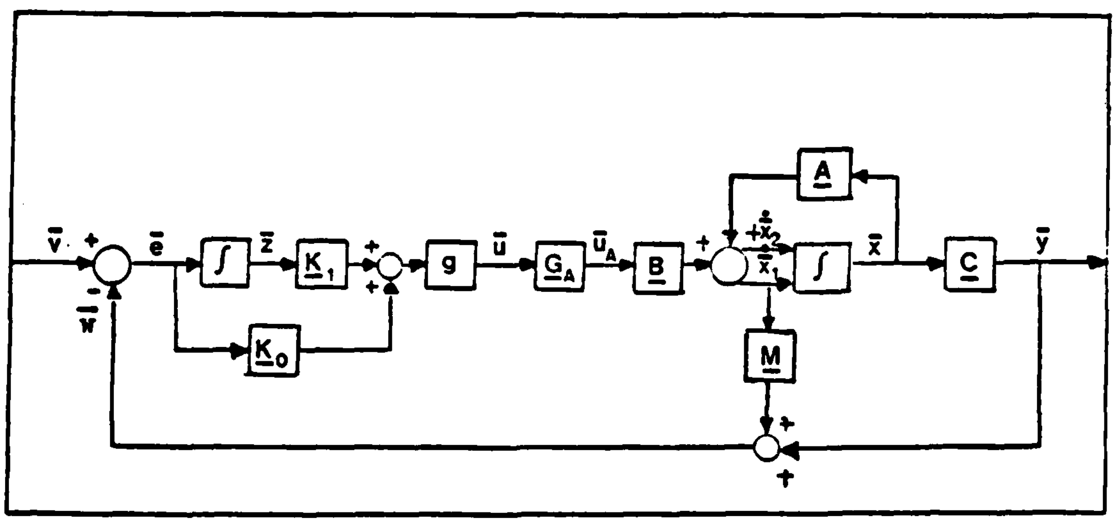


Figure 2-2 Tracking System with Actuators

III. Discussion of Results

Design Procedure

As a result of this thesis it is concluded that Porter's design method is easy to apply. Satisfactory controllers are found for a specific maneuver and flight condition without many design iterations. Also a designer's "feel" is quickly obtained, making controller designs easy to obtain. A feel is when the designer knows how different parameters affect the output response. The effect of the various parameters discovered during the design iterations for the controllers of this report are now discussed.

The measurement matrix, $\underline{M}$, is selected as a sparse matrix using the second method discussed in Chapter II (Ref 9). The reciprocal of the non-zero element of the $\underline{M}$ matrix becomes the transmission zero of the system. A transmission zero of -4 is selected for all of the controller designs in this report. This value is recommended by Lt Ridgely (Ref 10).

The value of the sigma matrix, $\underline{\Sigma}$, does affect the outputs in a predictable way. To keep the outputs decoupled, and the fast asymptotic transfer function, $\hat{\underline{\Gamma}}$, diagonal, $\underline{\Sigma}$ is selected as a diagonal matrix. Each diagonal element, σ_i, affects the corresponding output (the first diagonal element affects the first output, etc.). Increasing the value of the element causes the output to track the command input closer. A large diagonal sigma element is desirable when the specific output response is commanded to zero. An adverse effect of a large diagonal element is that the magnitude of the control surface responses are increased. Decreasing the diagonal element causes an

increase in the transient response of the output. Varying the diagonal elements in the sigma matrix is the main avenue used to achieve desired output and control surface time responses for the controller's designs contained in this report. An observation made during the design iterations is that an optimal sigma matrix found for a specific maneuver and specific flight condition is very close to the optimal sigma matrix for the same maneuver at a different design flight condition. This observation is shown by comparing the sigma matrices for the three yaw pointing designs, Tables 3-2 through 3-4.

The gain value, g, is used in the design iterations for the controllers in this report, to adjust the peak value and settling time of the response to a step input. The gain can be adjusted so the output response has no noticeable peak and an acceptable settling time. For all values of gain an overshoot is found to exist. Very high gain and very low gain cause a large overshoot. The gain for a minimum overshoot is a medium value. Settling time decreases with increasing gain value.

The value of the input commands can be used to adjust the control surface time responses. Changes to the values of the input commands do not adversely affect the output responses, in that they will track the commands. The control surface responses are, naturally, sensitive to the magnitude of the input command. The value of the input command can be reduced to keep the control surface deflections within limits. This is the last parameter to be adjusted in the design procedure. The initial values of input commands are obtained from AFTI wind tunnel data (Ref 12).

The control surface deflection limits for the designs of this report are:

$$\text{Rudder, } -35^\circ \le \delta_r \le 35^\circ$$

$$\text{Aileron, } -35^\circ \le \delta_a \le 35^\circ$$

$$\text{Cannard, } -30^\circ \le \delta_c \le 30^\circ$$

These limit values are selected by the designer and considered reasonable.

The interactive computer program, listed in Appendix C, make rapid design iterations possible.

Robustness Tests

As stated in the Introduction, the robustness tests are performed with the yaw pointing maneuver. The output matrix is

$$\left[C_1 \vdots C_2\right] = \begin{bmatrix} 0 & \vdots & 1 & 0 & 0 \\ 1 & \vdots & 0 & 0 & 0 \\ 0 & \vdots & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

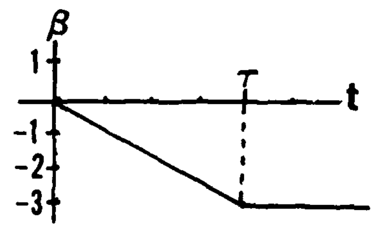
The values of the input commands to produce a yaw pointing angle are shown in Table 3-1. The side-slip angle, β, is commanded as a ramp with the slope $-\beta_o$ until the value of the side-slip angle reaches minus three degrees. At this time, τ, the slope is changed to zero. The ramp input is used as a way of commanding the time rate of the side-slip angle, $\dot{\beta}$. The roll angle, ϕ, is commanded to zero. The yaw rate, r, is commanded as a pulse with a constant magnitude equal to β_o. The value of β_o is a function of the dynamic pressure, Q, which is determined by the flight condition. At time τ the value of yaw rate is commanded to zero.

TABLE 3-1

INPUT COMMANDS

1) **Side Slip Angle, β**

$$\beta(t) = -\beta_0 t \quad 0 \leq t \leq \tau$$
$$= -3 \quad t > \tau$$

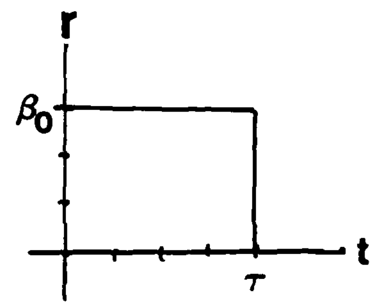


2) **Roll Angle, ϕ**

$$\phi(t) = 0 \quad t \geq 0$$

3) **Yaw Rate, r**

$$r(t) = \beta_0 \quad 0 \leq t \leq \tau$$
$$= 0 \quad t > \tau$$



Q CONDITION	SLOPE, β_0	TIME CONSTANT, τ	$\int_0^\tau \beta(t)dt$	$\int_0^\tau r(t)dt$
	(deg/sec)	(seconds)	(deg·sec)	(degrees)
Low	0.80	3.75	5.625	3.00
Low-Mid	1.00	3.00	4.500	3.00
Mid	3.00	1.00	1.500	3.00
High	4.00	0.75	1.125	3.00

The results of the yaw pointing robustness test for varying flight conditions are summarized in Tables 3-2 through 3-4 and Figures 3-2a through 3-2f.* In all cases, except one (Design 1A, Q = 825) the controller performs the desired yaw pointing maneuver. This is shown by the fact that the steady-state side-slip angle, β_{ss}, and the steady-state pointing angle, ψ_{ss}, are equal in magnitude and opposite in sign. The pointing angle, ψ, is the time integral of the yaw rate

$$\psi(t) = \int_0^t r(t)dt \tag{3.2}$$

The most noticeable difference, when the controller is used outside its design flight condition, is the response of the yaw rate, r. Outside the design flight condition the yaw rate has an initial overshoot and a few oscillations. This does not affect the performance of the maneuver, based on the smooth response of the side-slip angle and the pointing angle.

Figures 3-3 through 3-5 show the output and control surface time responses of all three controllers performing a yaw pointing maneuver at a specific flight condition. Figure 3-3 shows the responses of a low dynamic pressure maneuver (Q = 109). Figure 3-4 shows the responses of a medium dynamic pressure maneuver (Q = 443). Figure 3-5 shows the responses of a high dynamic pressure maneuver (Q = 825).

The controller designed at a low dynamic pressure, Design 1A, consistently overcommanded the control surfaces when performing the yaw pointing maneuver outside its design flight condition, as shown

*Figure 3-1 shows the basis for the data in Tables 3-2 through 3-4.

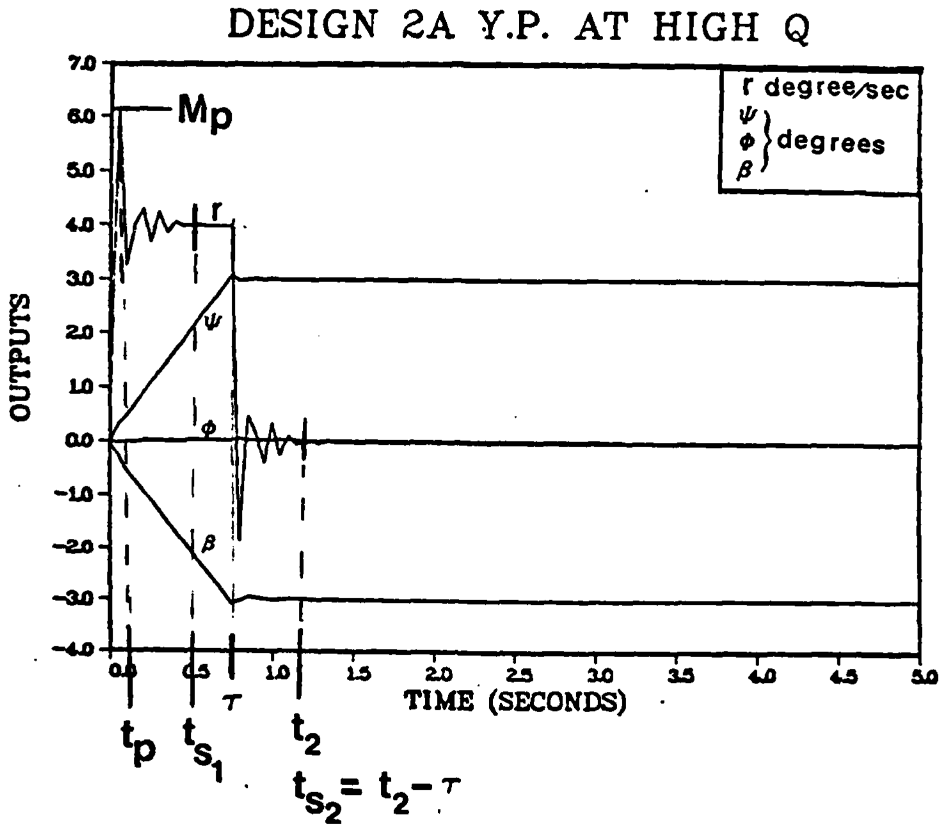
in Figure 3-2d. Controllers designed at medium dynamic pressure, Design 3A, only overcommanded the control surfaces at two of the ten flight conditions simulated, and the value of the deflection error is not great.

Figures 3-2e and 3-2f show the range of motion of the velocity vector pointing angle. The velocity vector is the sum of the sideslip angle and the pointing angle. Figure 3-2e shows the range of values that the velocity vector pointing angle moves. In the worst case (Design 1A, Q = 825) the total horizontal translation is not greater than 10 feet, during the time of the maneuver, τ. In this case the steady-state velocity vector pointing angle is 0.13 degrees, resulting in a steady-state horizontal velocity of 2.2 feet/second.

Figure 3-2c shows the time integral of the roll angle. This quantity is used as a judgement of performance, and a basis for comparison. In the worst case (Design 3A, Q = 70) the value is 0.176 degree-second. This is equivalent to a roll angle of 0.06 degrees for three seconds. The magnitudes of the roll angle time integral shows that in all cases the plane remained essentially wings level.

Based on all performance criteria shown in Tables 3-2 through 3-4, and Figures 3-2a through 3-2f, the controller designed for a medium dynamic pressure flight condition, Design 2A, is the more robust design.

Another robustness test is judging the performance of a controller designed to perform a specific maneuver at a specific flight condition performing other maneuvers at the same flight condition.



$$\%\,\text{error} = \frac{Mp - \beta_0}{\beta_0}\,100$$

Figure 3-1 Figures of Merit for Output Responses

TABLE 3-2 Design 1A Figures of Merit

DESIGN 1A LOW Q $Q = 109\ lb/ft^2$ $h = 20,000$ ft Mach = 0.4

$$\underline{M} = \begin{bmatrix} 0 \\ 0.25 \\ 0 \end{bmatrix} \quad \underline{\Sigma} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 30 \end{bmatrix} \quad \underline{A}_0 = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix}$$

$$\underline{K}_0 = \begin{bmatrix} 29.1768 & 1.6411 & -6.4843 \\ 5.4674 & -4.3414 & -1.6377 \\ 44.6171 & 0.5608 & 20.4908 \end{bmatrix} \quad \underline{K}_1 = \underline{K}_0 \quad g = 29.5$$

Q	β_{ss}	$\int_0^t r(\xi)d\xi$ ψ_{ss}	M_p	%error	t_p	t_{s_1}	t_{s_2}	$\int_0^t \phi(\xi)d\xi$	CONTROL SURFACE COMMENT
70	-2.999	2.993	1.0508	31.350	0.19	0.40	0.45	0.0414	exceeds limits $\delta c > 38.0°$
109	-2.999	3.012	0.8197	2.462	0.35	0.40	0.30	0.0242	within limits
163	-2.999	3.005	1.0114	28.453	0.10	0.45	0.40	0.0162	exceeds limits $\delta c > 10.9°$
223	-2.999	2.992	1.5377	53.770	0.05	0.40	0.45	0.0188	within limits
436	-2.999	3.018	3.7456	24.853	0.10	0.45	0.35	0.0166	exceeds limits $\delta c > 28.9°$
443	-2.999	3.037	3.6850	22.820	0.15	0.45	0.35	0.0147	exceeds limits $\delta c > 43.1°$
532	-2.999	3.007	4.2513	41.710	0.05	0.40	0.40	0.0153	exceeds limits $\delta c > 28.1°$
652	-2.999	2.9915	4.5565	51.883	0.05	0.40	0.40	0.0164	exceeds limits $\delta c > 33.9°$
825	-2.999	3.132	4.2150	5.375	0.60	0.75	0.10	0.0159	exceeds limits $\delta r > 10.6°$
1198	-2.999	2.993	6.3388	58.470	0.05	0.40	0.15	0.0171	exceeds limits $\delta r > 6.5°$

TABLE 3-3 Design 2A Figures of Merit

DESIGN 2A MID Q $Q = 443\ lb/ft^2$ $h = 5{,}000$ Mach $= 0.6$

$$\underline{M} = \begin{bmatrix} 0 \\ 0.25 \\ 0 \end{bmatrix} \qquad \underline{\Sigma} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 30 \end{bmatrix} \qquad \underline{A}_{\sigma} = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix}$$

$$\underline{K}_0 = \begin{bmatrix} 11.6944 & 0.1981 & -2.0615 \\ 2.7389 & -1.0522 & -0.1395 \\ 14.6517 & -0.2256 & 4.4020 \end{bmatrix} \qquad \underline{K}_1 = 2\underline{K}_0 \qquad g = 29.0$$

Q	β_{ss}	$\int_0^t r(\xi)d\xi$ ψ_{ss}	M_p	%error	t_p	t_{s_1}	t_{s_2}	$\int_0^t \phi(\xi)d\xi$	CONTR SURFA COMME
70	-3.000	3.011	1.1771	47.1375	0.05	0.50	0.45	0.1069	within limits
109	-3.000	3.001	1.3366	67.0750	0.05	0.45	0.45	0.0644	within limits
163	-3.000	2.995	1.1883	48.5250	0.05	0.45	0.45	0.0459	within limits
223	-3.000	2.9821	1.4072	40.7200	0.10	0.45	0.40	0.0463	within limits
436	-3.000	3.053	3.0801	2.6700	0.40	0.30	0.30	0.0385	within limits
443	-3.000	3.043	3.0414	1.3800	0.40	0.20	0.35	0.0342	within limits
532	-3.000	3.020	3.6418	21.3933	0.15	0.40	0.40	0.0324	within limits
652	-3.000	3.011	4.1880	39.6000	0.10	0.50	0.35	0.0368	within limits
825	-3.000	2.993	6.0860	52.1500	0.05	0.40	0.40	0.0367	exceeds limits $\delta_c > 2.0$
1198	-3.000	2.981	5.4505	36.2625	0.05	0.45	0.40	0.0365	exceeds limits $\delta_c > 5.0$

TABLE 3-4 Design 3A Figures of Merit

DESIGN 3A HIGH Q $Q = 825\ lb/ft^2$ $h = 10{,}000$ ft Mach = 0.9

$$\underline{N} = \begin{bmatrix} 0 \\ 0.25 \\ 0 \end{bmatrix} \quad \underline{\Sigma} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 30 \end{bmatrix} \quad \underline{A}_a = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix}$$

$$\underline{K}_0 = \begin{bmatrix} 22.5871 & 0.1857 & -1.6393 \\ 5.9893 & -0.7110 & -0.0607 \\ 22.3193 & -0.1024 & 1.9477 \end{bmatrix} \quad \underline{K}_1 = 2\underline{K}_0 \quad g = 29.5$$

Q	β_{ss}	$\int_0^t r(\xi)d\xi \,/\, \psi_{ss}$	M_p	%error	t_p	t_{s_1}	t_{s_2}	$\int_0^t \phi(\xi)d\xi$	CONTROL SURFACE COMMENT
70	-3.000	3.045	1.1625	45.312	0.10	0.80	0.60	0.1763	within limits
109	-3.000	3.025	1.2601	57.512	0.05	0.50	0.55	0.1119	within limits
163	-3.000	3.021	1.3857	73.212	0.05	0.50	0.45	0.0829	within limits
223	-3.000	3.009	1.6871	68.710	0.05	0.45	0.50	0.1361	within limits
436	-3.000	2.972	4.1703	39.010	0.10	0.50	0.60	0.0705	exceeds limits $\delta c > 8.6°$
443	-3.000	2.952	4.0944	36.480	0.10	0.40	0.65	0.0662	exceeds limits $\delta c > 2.3°$
532	-3.000	2.907	3.6494	21.646	0.15	0.95	0.60	0.0632	within limits
652	-3.000	2.951	3.0646	2.153	0.30	0.40	0.55	0.0652	within limits
825	-3.000	3.041	4.1046	2.615	0.35	0.45	0.60	0.0635	within limits
1198	-3.000	3.045	5.4566	36.415	0.10	0.40	0.55	0.0655	within limits

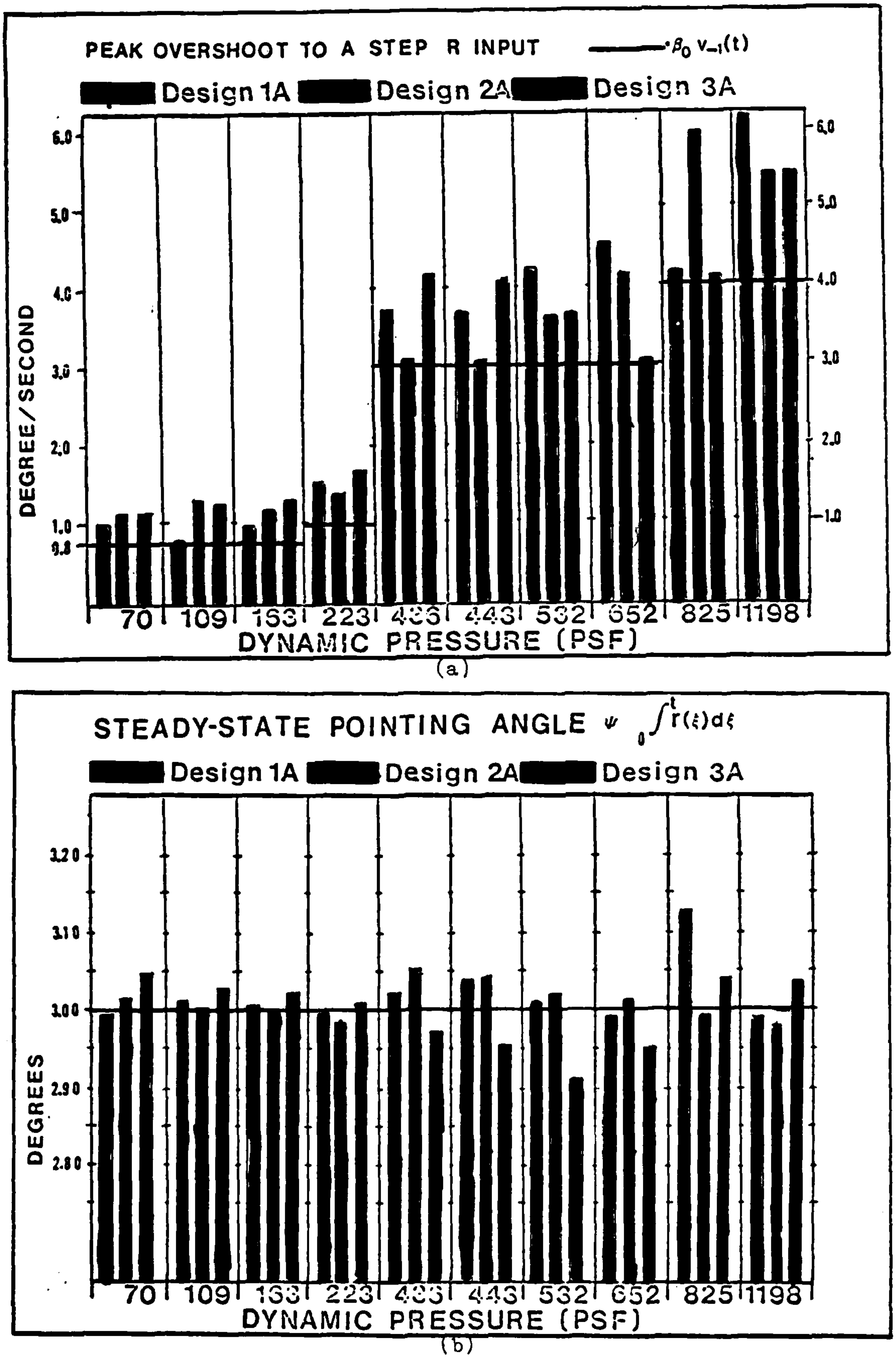
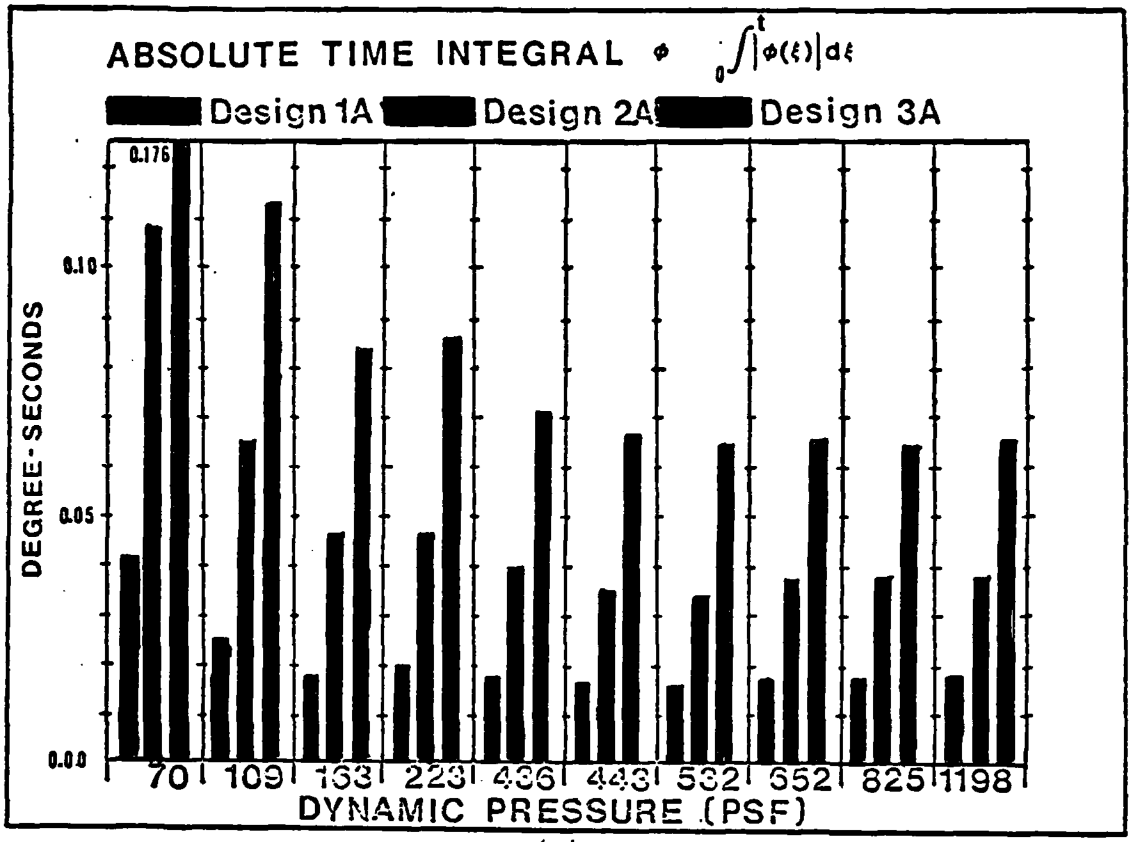
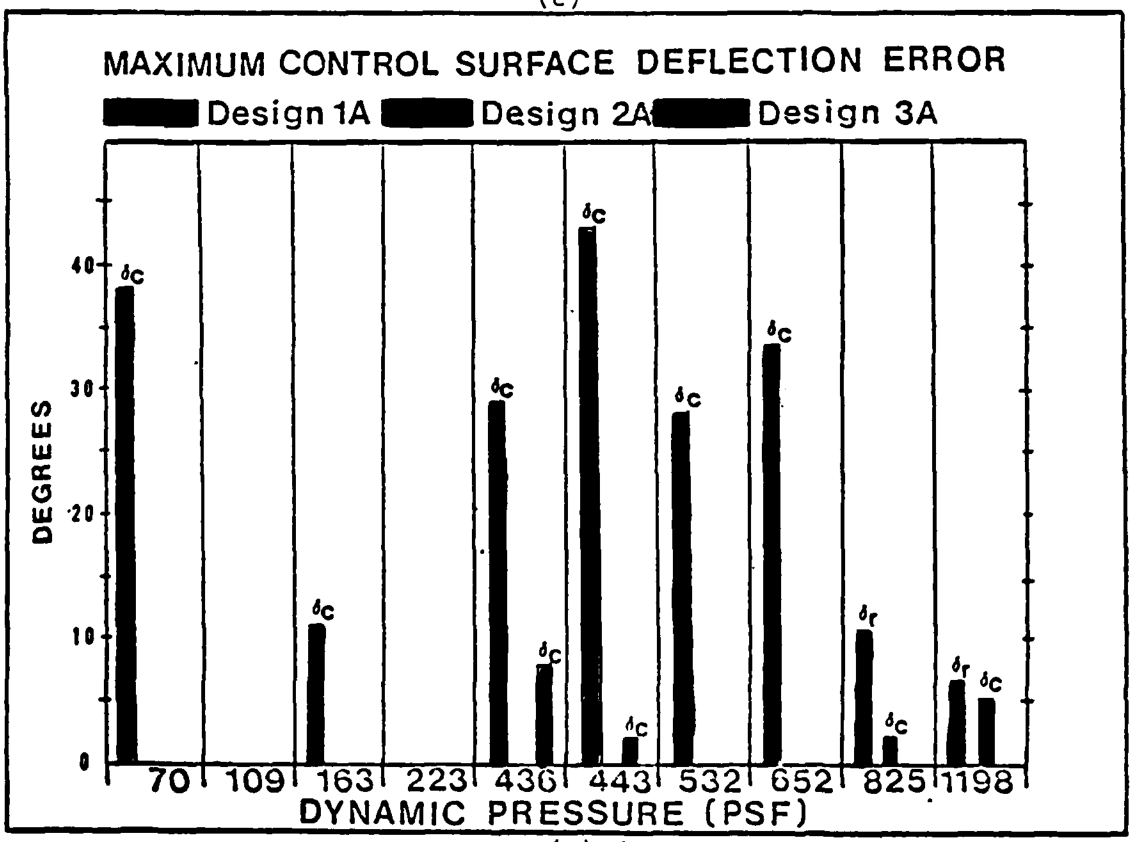


Figure 3-2 Comparison of Figures of Merit



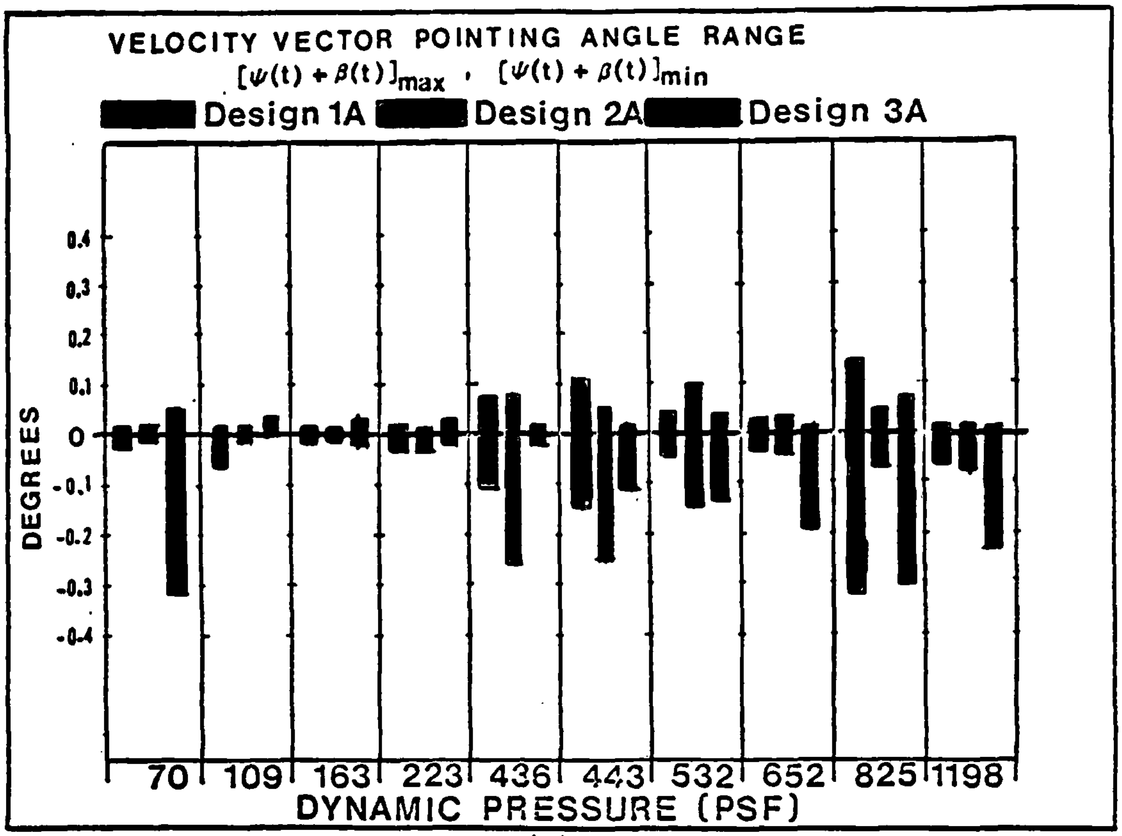
ABSOLUTE TIME INTEGRAL ϕ $\int_0^t |\phi(\xi)| d\xi$
Design 1A
Design 2A
Design 3A
DEGREE - SECONDS
0.176
0.10
0.05
0.00
70
109
163
223
436
443
532
652
825
1198
DYNAMIC PRESSURE (PSF)

(c)

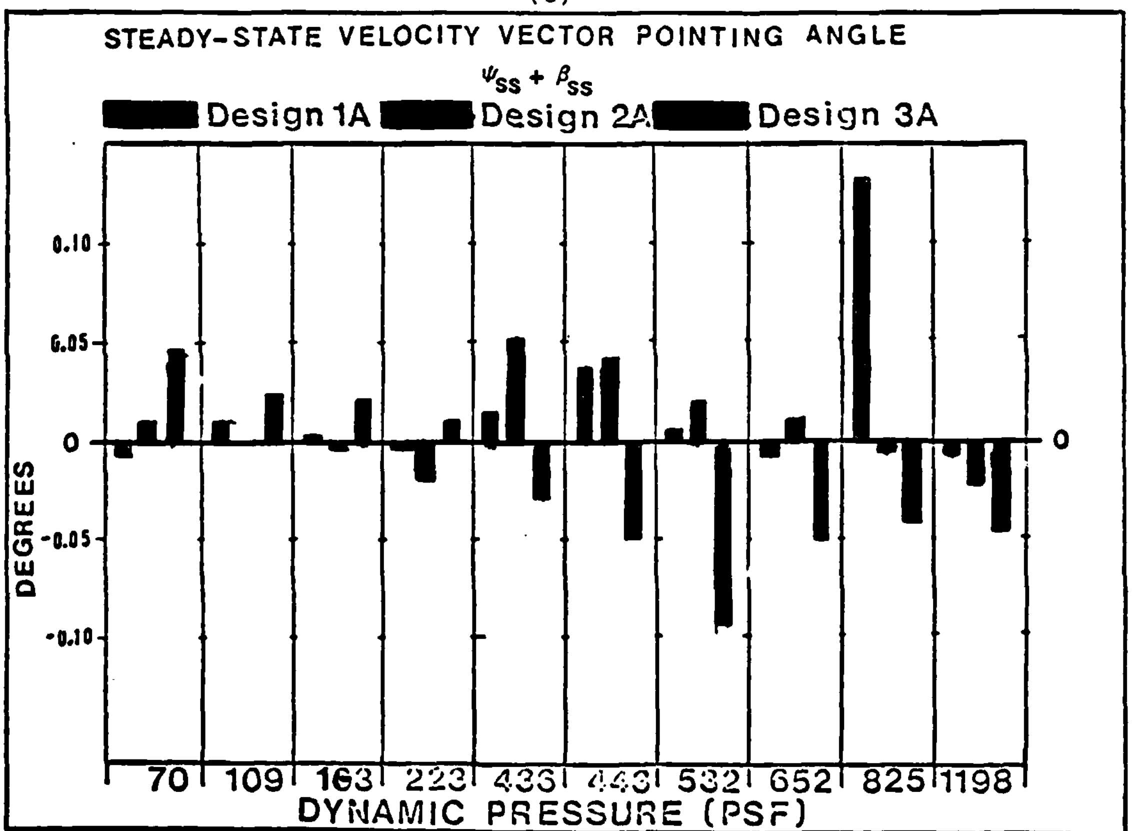


MAXIMUM CONTROL SURFACE DEFLECTION ERROR
Design 1A
Design 2A
Design 3A
DEGREES
40
30
20
10
0
δ_c
δ_r
70
109
163
223
436
443
532
652
825
1198
DYNAMIC PRESSURE (PSF)

(d)


VELOCITY VECTOR POINTING ANGLE RANGE
$[\psi(t) + \beta(t)]_{max}$, $[\psi(t) + \beta(t)]_{min}$
Design 1A
Design 2A
Design 3A
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
DEGREES
70
109
163
223
436
443
532
652
825
1198
DYNAMIC PRESSURE (PSF)

(e)


STEADY-STATE VELOCITY VECTOR POINTING ANGLE
$\psi_{ss} + \beta_{ss}$
Design 1A
Design 2A
Design 3A
0.10
0.05
0
-0.05
-0.10
DEGREES
0
70
109
163
223
436
443
532
652
825
1198
DYNAMIC PRESSURE (PSF)

(f)

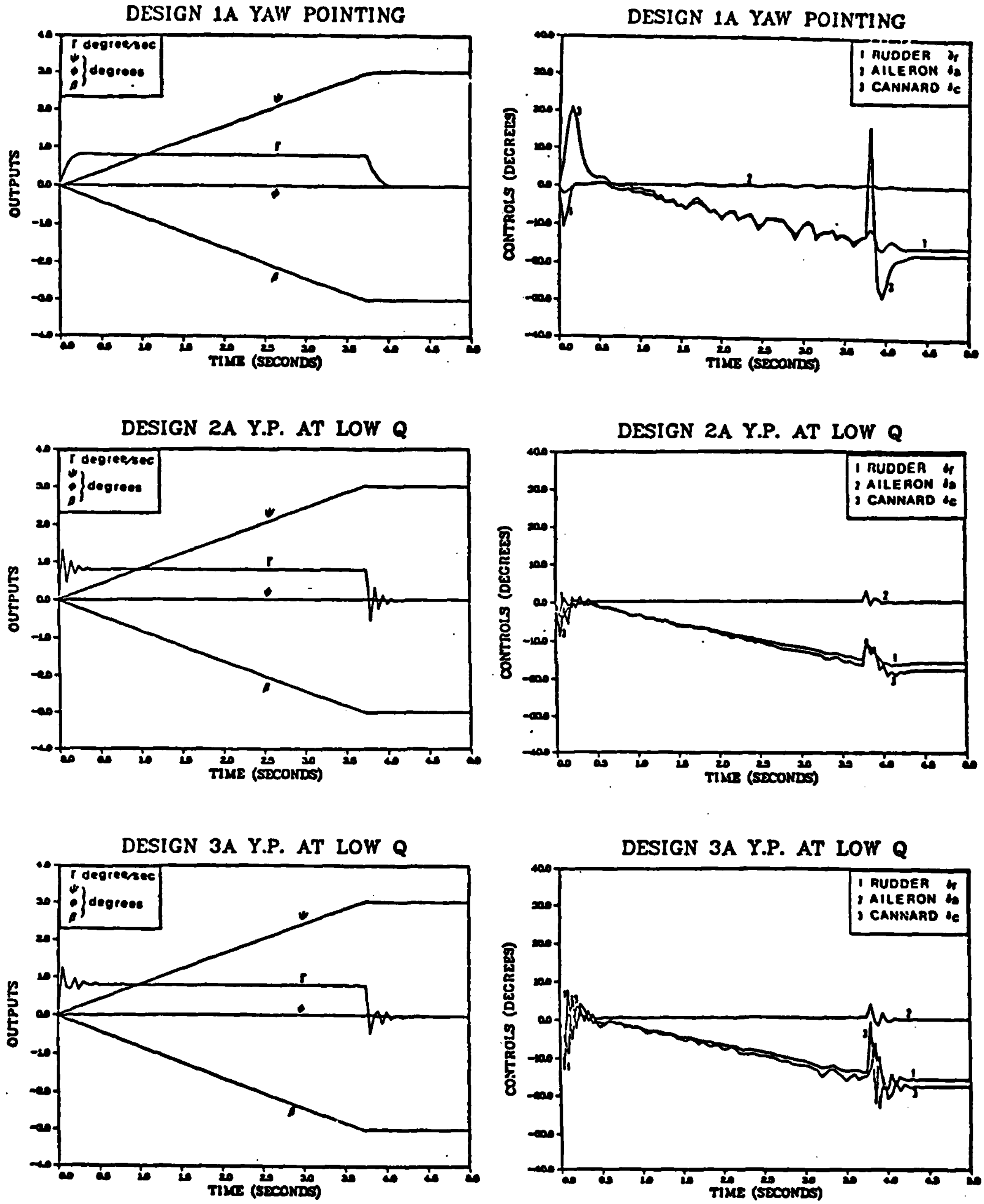


Figure 3-3 Low Q Yaw Pointing

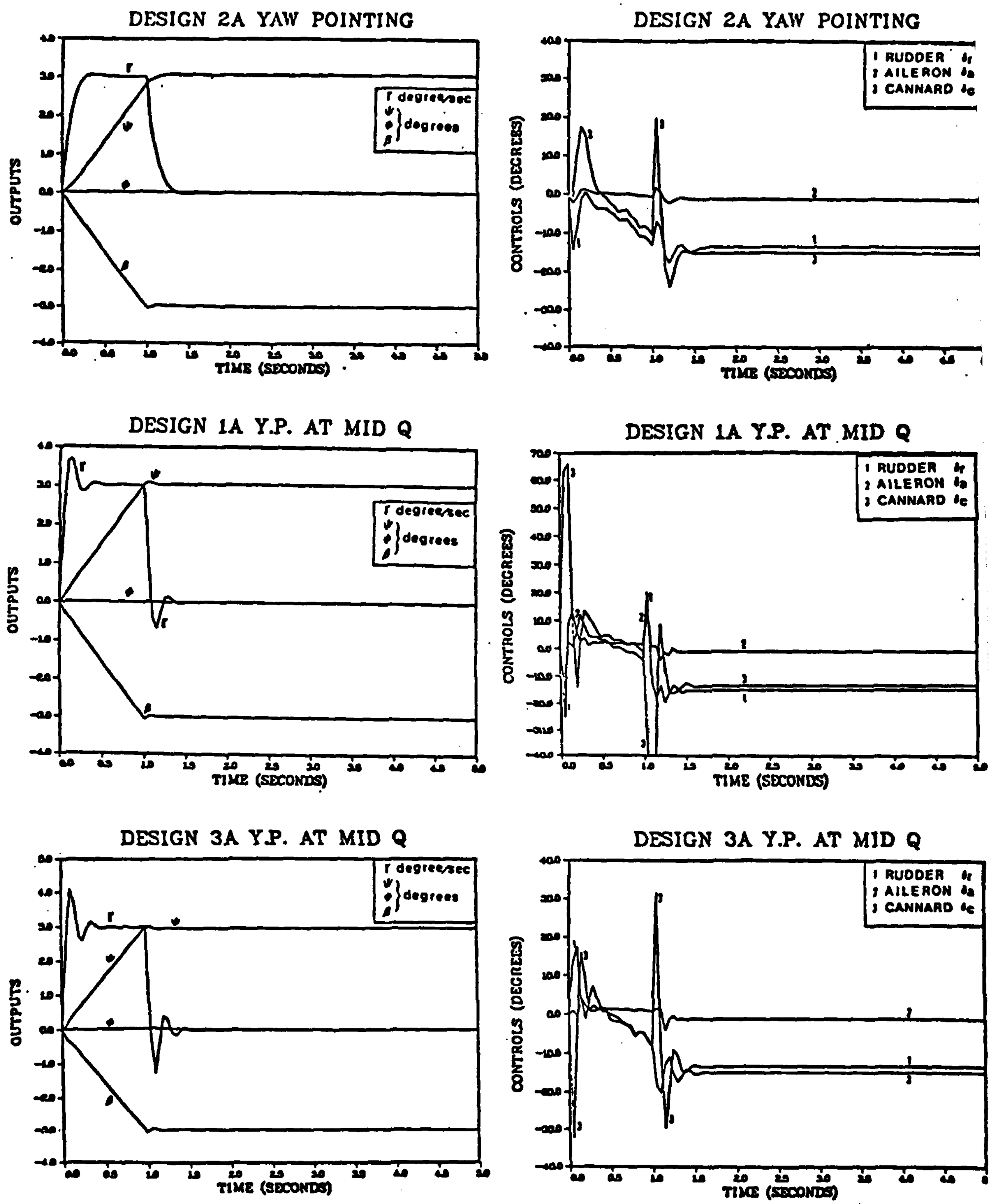


Figure 3-4 Mid Q Yaw Pointing

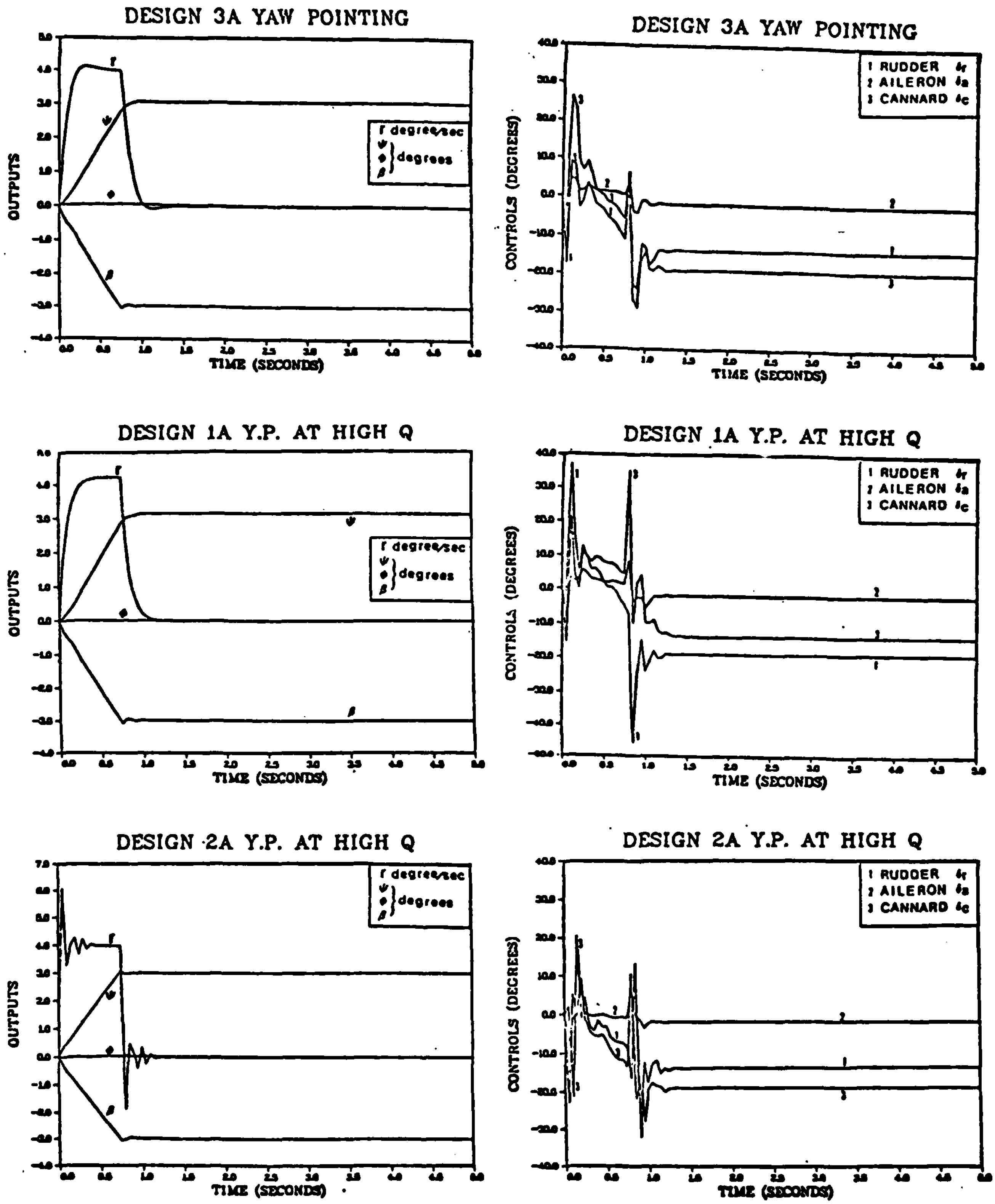


Figure 3-5 High Q Yaw Pointing

The controller design 2A is given commands to perform a wings level turn, a horizontal translation, and a constant altitude coordinated turn. The magnitudes of these commands are adjusted until the maneuver can be performed without commanding the control surfaces beyond their limits.

The input command for a wings level turn is a step input for yaw rate, r, while side-slip angle, β, and roll angle, ϕ, are commanded to zero. For horizontal translation the side-slip angle is commanded as a ramp, till a side-slip angle of four degrees is reached. The roll angle and yaw rate are commanded to zero. For a coordinated turn the relation between yaw rate and roll angle is

$$r = \frac{g}{U} \sin \phi \tag{3.3}$$

while side-slip angle is commanded to zero.

The output and control surface time responses of Design 2A perfoming the above three maneuvers are shown in Figure 3-6. These responses are compared with those of Figure 3-7 which show the responses of controllers designed to specifically perform the maneuvers at the design 2A flight condition (Q = 443). It is easily seen that Design 2A can not perform the maneuvers as well as the design for that specific maneuver. The magnitudes of the maneuvers of Design 2A are much less than the magnitudes of the Mid Q designs. For example the Mid Q design for horizontal translation can successfully command a side-slip angle of four degrees in 1.33 seconds, while the controller of Design 2A took 4.5 seconds. In this respect, the controller is not robust with respect to varying command inputs. The Mid Q

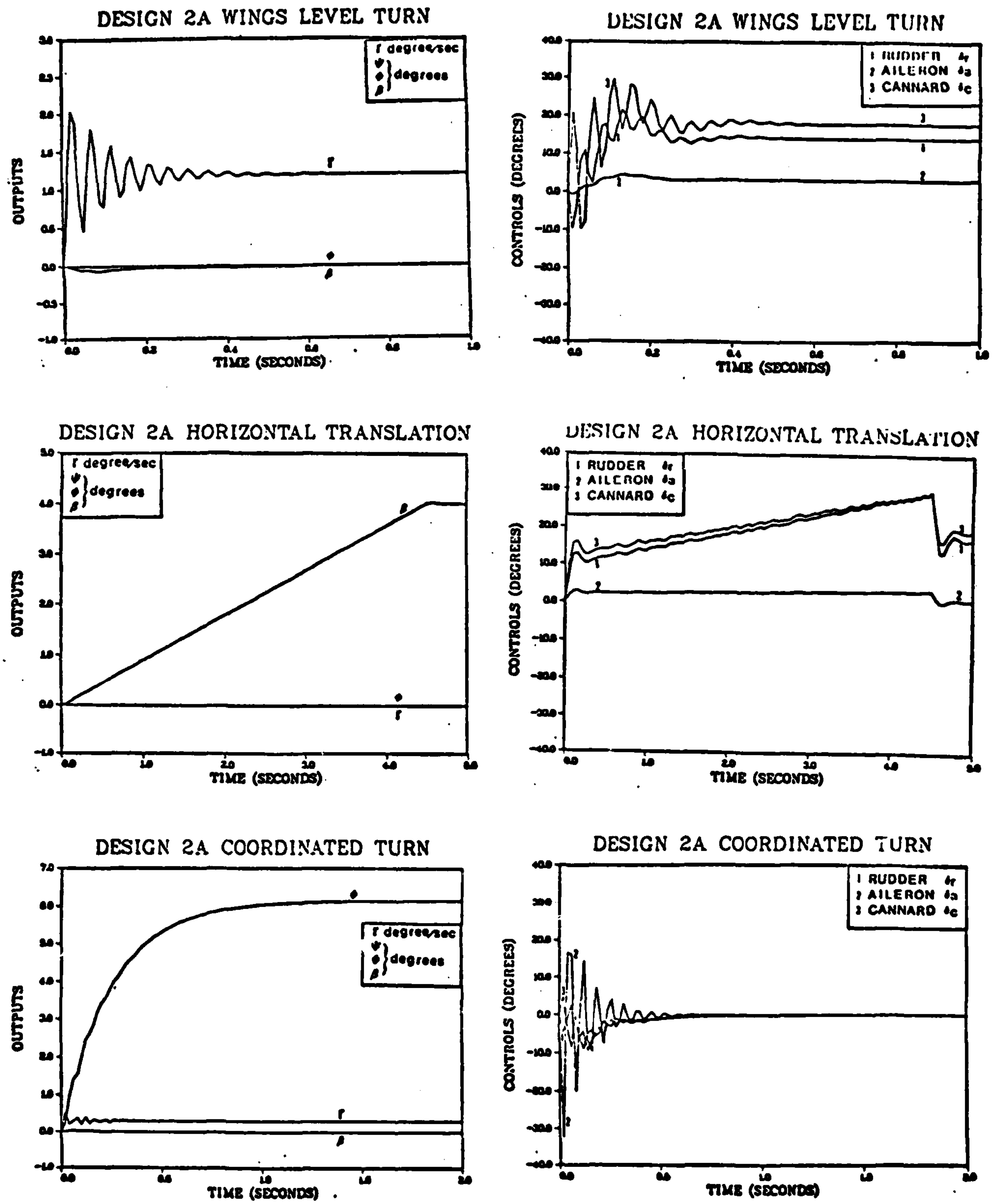


Figure 3-6 Design 2A Additional Maneuvers

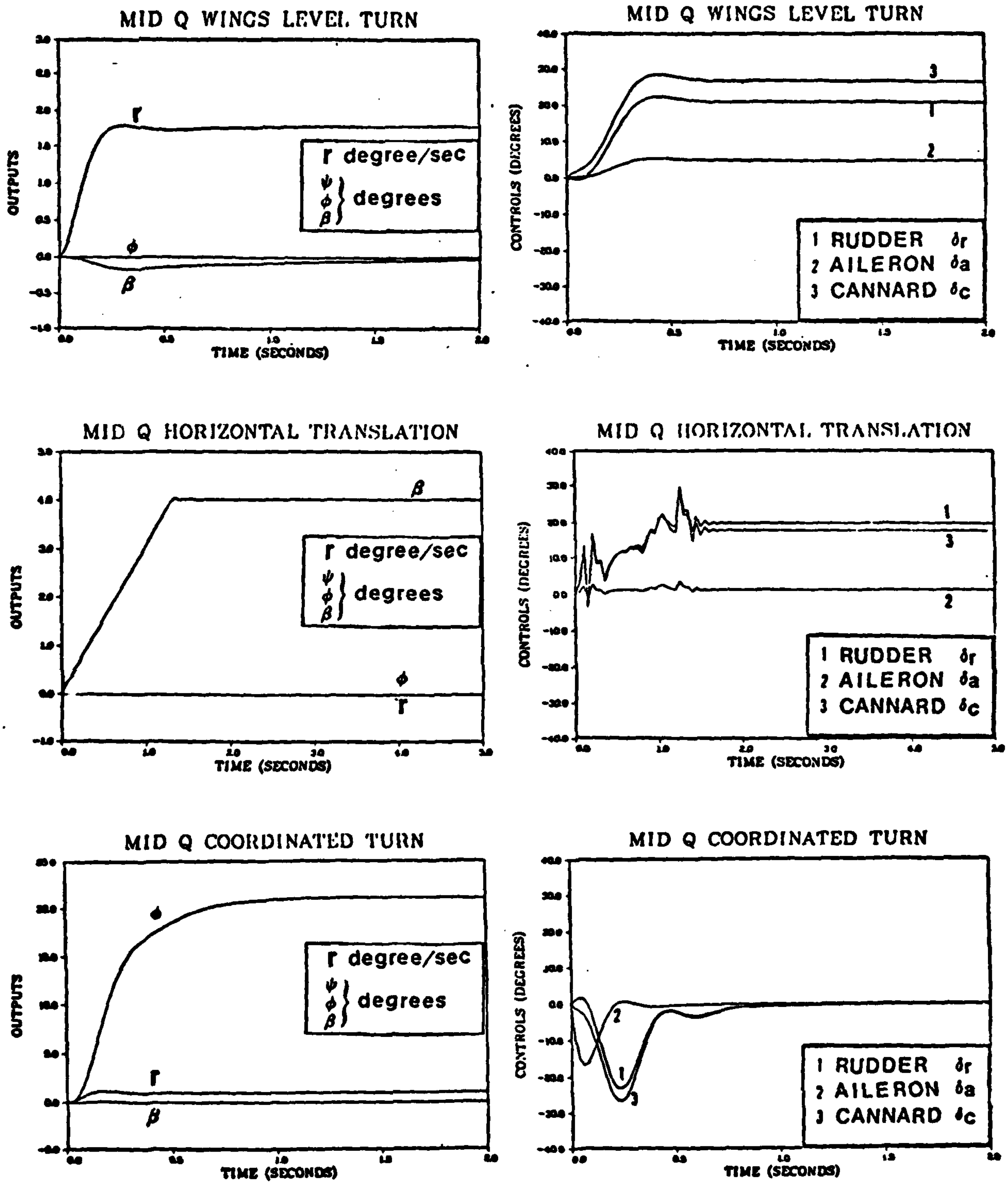


Figure 3-7 Mid Q Additional Maneuvers

Table 3-5 Mid Q designs

MID Q WINGS LEVEL TURN

$$\underline{M} = \begin{bmatrix} 0 \\ 0.25 \\ 0 \end{bmatrix} \qquad \underline{\Sigma} = \begin{bmatrix} .5 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & .5 \end{bmatrix} \quad \underline{A}_c = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix}$$

$$\underline{K}_0 = \begin{bmatrix} 5.8472 & 0.1321 & -0.0344 \\ 1.3695 & -0.7015 & -0.0023 \\ 7.3259 & -0.1504 & 0.0734 \end{bmatrix} \quad \underline{K}_1 = \underline{K}_0 \qquad g = 20$$

	Steady-state Yaw Rate	Peak Overshoot	Settling Time
Desgign 2A	1.20 deg/sec	2.0 deg/sec	0.5 sec
Mid Q	1.75 deg/sec	1.8 deg/sec	0.5 sec

MID Q HORIZONTAL TRANSLATION

$$\underline{M} \quad \begin{bmatrix} 0 \\ 0.25 \\ 0 \end{bmatrix} \qquad \underline{\Sigma} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 75 \end{bmatrix} \quad \underline{A}_c = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix}$$

$$\underline{K}_0 = \begin{bmatrix} 58.4719 & 0.6604 & -5.1539 \\ 13.6946 & -3.5073 & -0.3487 \\ 73.2587 & -0.7521 & 11.0051 \end{bmatrix} \quad \underline{K}_1 = 2\underline{K}_0 \qquad g = 40$$

	Steady-state Side-slip angle	Maneuver Time	Maneuver Rate
Design 2A	4.00 degrees	4.50 seconds	0.9 deg/sec
Mid Q	4.00 degrees	1.33 seconds	3.0 deg/sec

MID Q COORDINATED TURN

$$\underline{M} = \begin{bmatrix} 0 \\ 0.25 \\ 0 \end{bmatrix} \qquad \underline{\Sigma} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \underline{A}_c = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix}$$

$$\underline{K}_0 = \begin{bmatrix} 11.6944 & 0.0132 & -0.0697 \\ 2.7389 & -0.0701 & -0.0046 \\ 14.6517 & -0.0150 & 0.1457 \end{bmatrix} \quad \underline{K}_1 = \underline{K}_0 \qquad g = 20$$

	Steady-state Roll Angle	Steady-state Yaw Rate
Design 2A	6.15 degrees	0.30 deg/sec
Mid Q	20.93 degrees	1.00 deg/sec

controller designs and performance comparison summary are shown in Table 3-5.

Actuator Dynamics

As discussed in Chapter 2 there are two methods to add first-order actuator dynamics to the system. The method used in this report is to augment the closed-loop system matrix. This does not cause the size of the measurement matrix to increase. If the actuator dynamics are added by augmenting the open-loop system matrix, the size of the measurement matrix increases from a 3-by-1 matrix to a 3-by-4 matrix. When an attempt is made to add the actuator dynamics this way, the method used for selecting a measurement matrix that both stabilizes the system and diagonalizes the asymptotic transfer function fails. This leaves the selection of a measurement matrix to a trial and error iterative procedure.

The actuator dynamics are significant in the simulations. For comparison the yaw pointing robustness test is repeated for the same three controllers without actuator dynamics in the model. The results of these simulation tests are very misleading. Figures 3-8 through 3-10 show the output and control surface time responses of the three controller designs performing the yaw pointing maneuver at a specific flight condition. When these figures are compared with Figures 3-3 through 3-5 the significance of the actuator dynamics is clearly shown. Note that in all cases the outputs follow the commands exactly, with no overshoots or oscillations. Also note the smooth control surface response curves. The actuator dynamics in the model greatly affect the time responses and should not be ignored

when designing a controller. This same trend is observed when Figure 3-11 is compared with Figure 3-6.

Without actuator dynamics the output responses are much more insensitive to changes in the diagonal elements in the sigma matrix and gain values. This is another reason why the actuator dynamics should not be ignored when designing a controller.

Also studied is the effect of changing the first-order actuator dynamics time constant. The controller is designed with all actuator constants at 20. This design is used in the simulation of a yaw pointing maneuver with the actuator constants varying. The responses are shown in Figures 3-12 through 3-18. The three numbers on the title of the response correspond to the three actuator constants. The respective order is: rudder actuator constant, aileron actuator constant, and cannard actuator constant. For example, (20, 10, 40) represents a rudder constant of 20, a aileron constant of 10, and a cannard constant of 40.

Figure 3-12 shows the responses when all constants are decreased and increased by a factor of two. Figure 3-13 shows the responses when just one constant is decreased by a factor of two. Figure 3-14 shows the responses when just one constant is increased by a factor of two. Figure 3-15 shows the responses when two constants are decreased by a factor of two. Figure 3-16 shows the responses when two constants are increased by a factor of two. Figure 3-17 shows the responses when one constant is decreased and one constant is increased by a factor of two.

This actuator study shows:

1. The aileron has little effect in the yaw pointing maneuver

2. Decreasing the cannard constant increases settling time and increases cannard activity

3. Increasing the rudder constant causes increased rudder activity

4. Increasing both rudder and cannard constants cause the magnitude of surface deflection to decrease.

Overall this study amplifies the importance of actuator dynamics in the model, in that they significantly affect the output and control surface time responses. The actuator time constant is dependent on the physical system used in the aircraft. Varying the constants in the design can be advantageous in a preliminary design study and can be used as a basis for selecting the best actuator.

In this study there is no constraint placed on the rate of control surface deflection. As a result the control surface deflection rates were often unreasonably fast. Any further study should include this constraint.

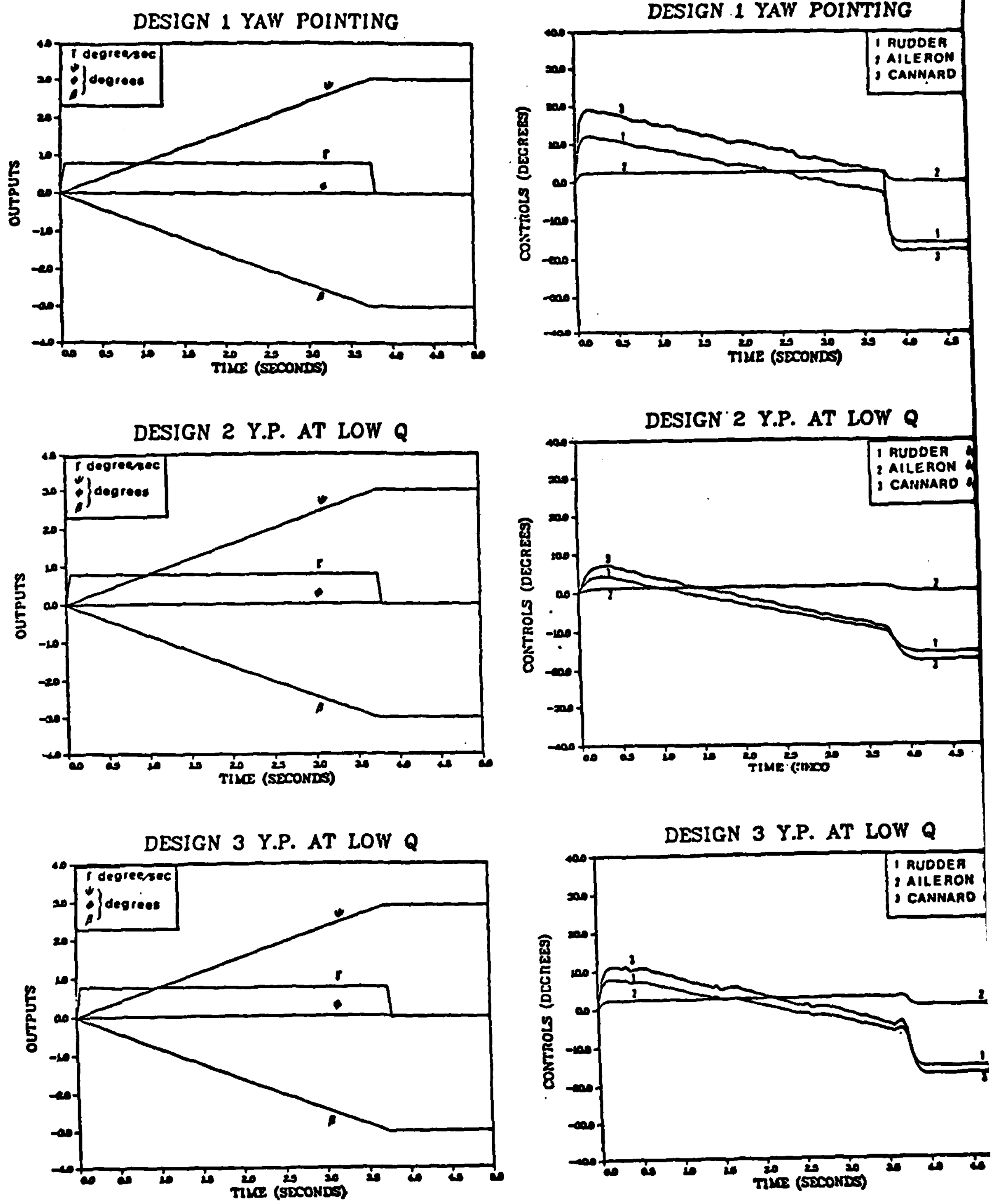


Figure 3-8 Low Q Yaw Pointing Without Actuators

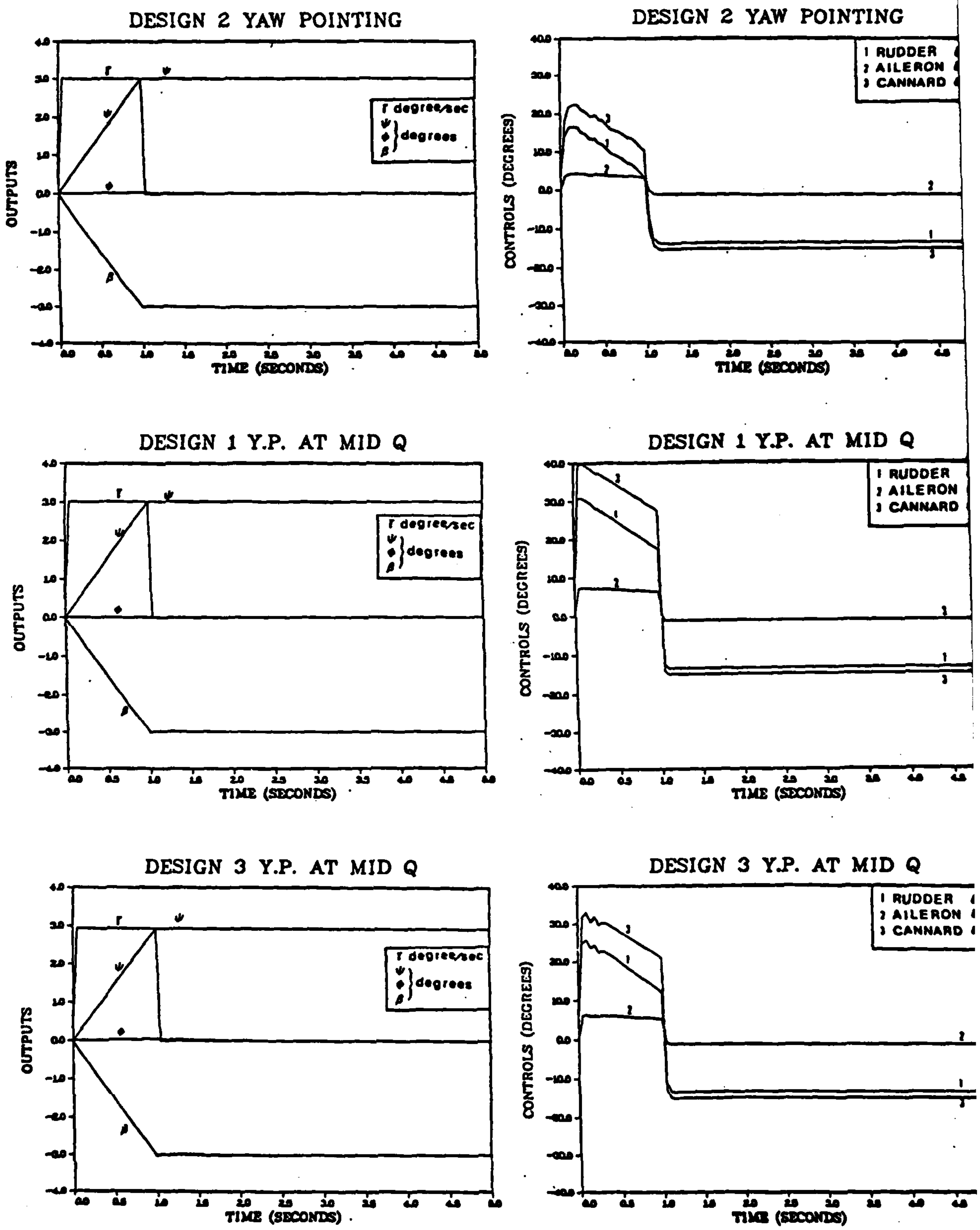


Figure 3-9 Mid Q Yaw Pointing Without Actuators

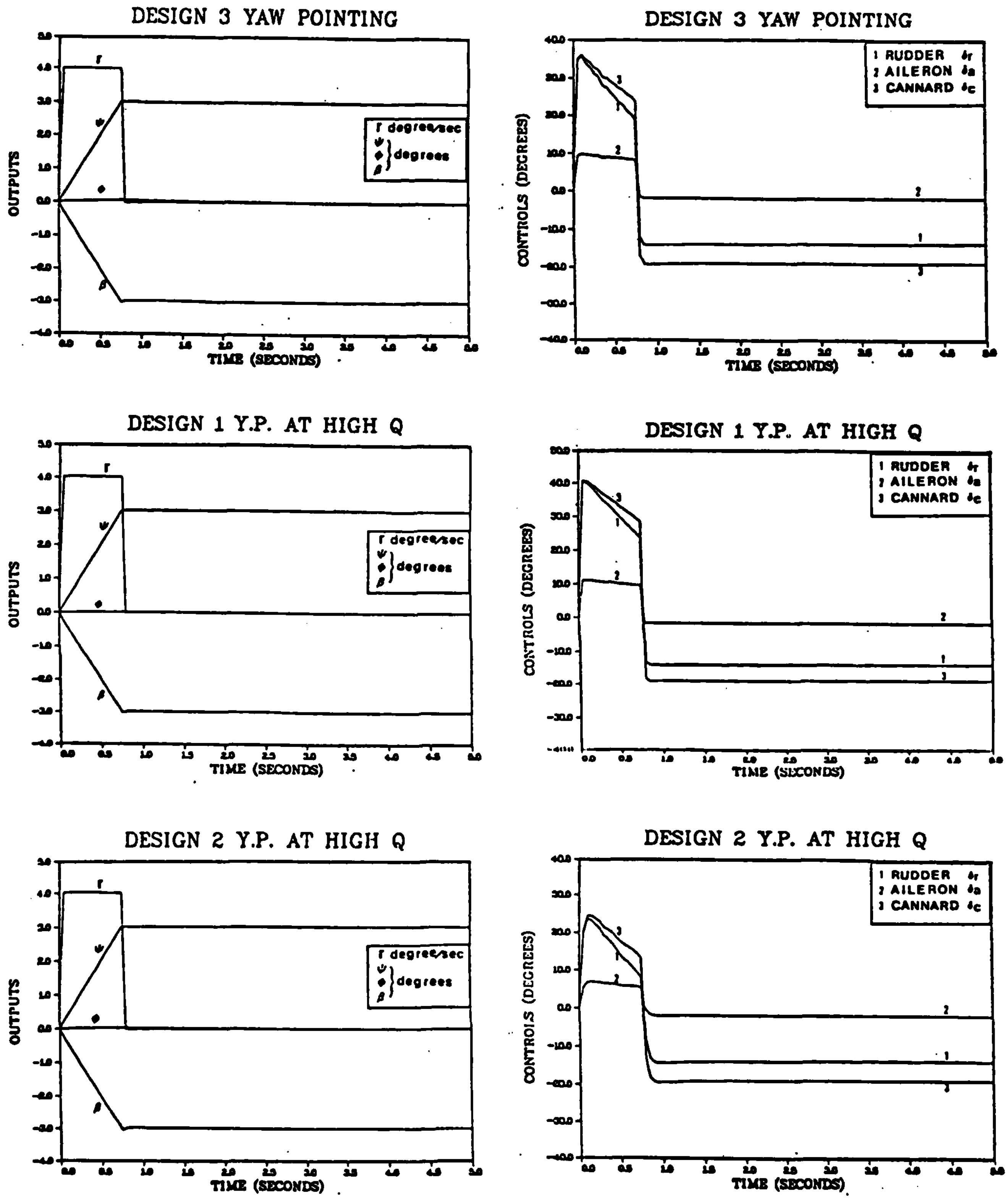


Figure 3-10 High Q Yaw Pointing Without Actuators

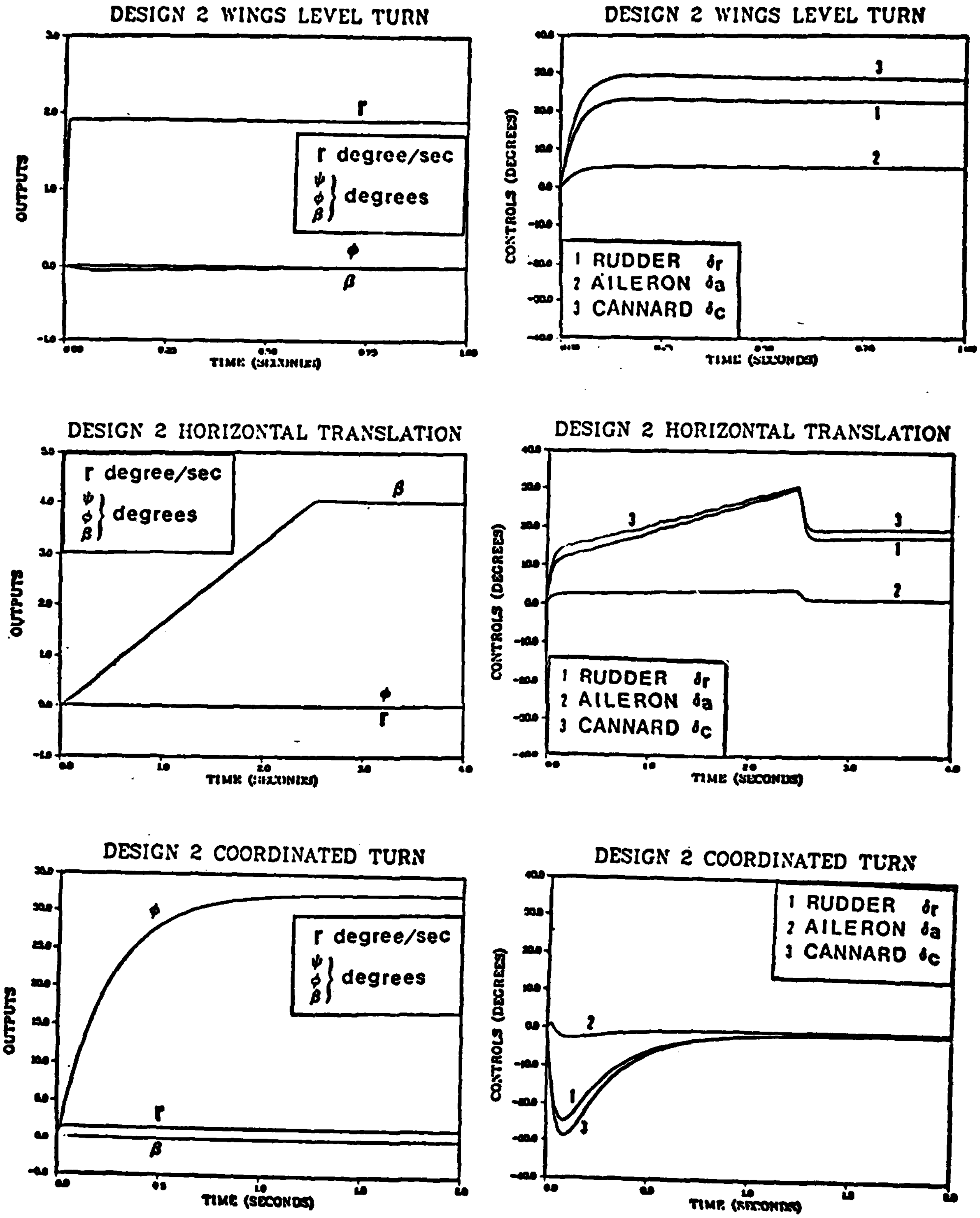


Figure 3-11 Design 2 Additional Maneuvers

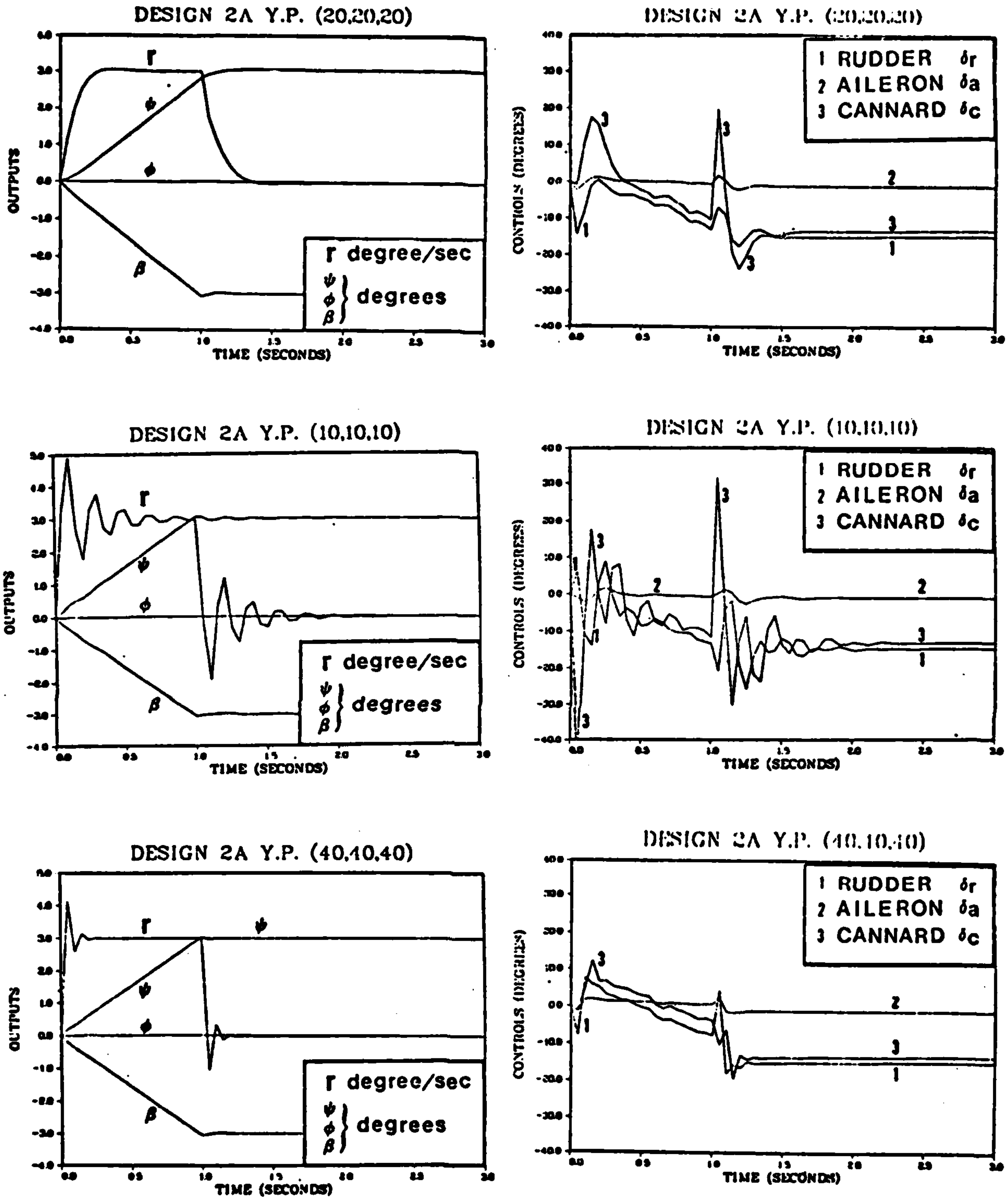


Figure 3-12 All Constants Changed

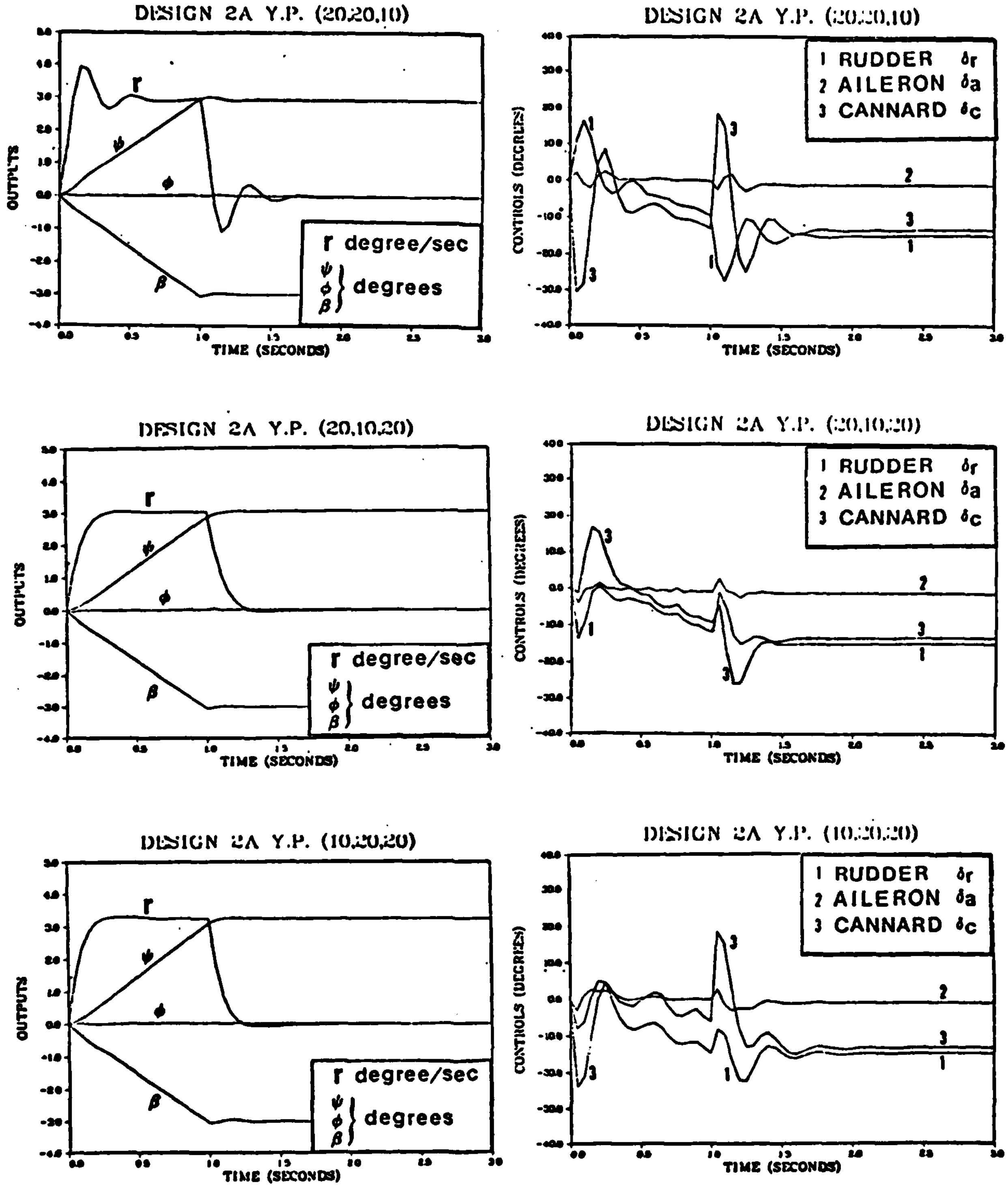


Figure 3-13 One Constant Decreased

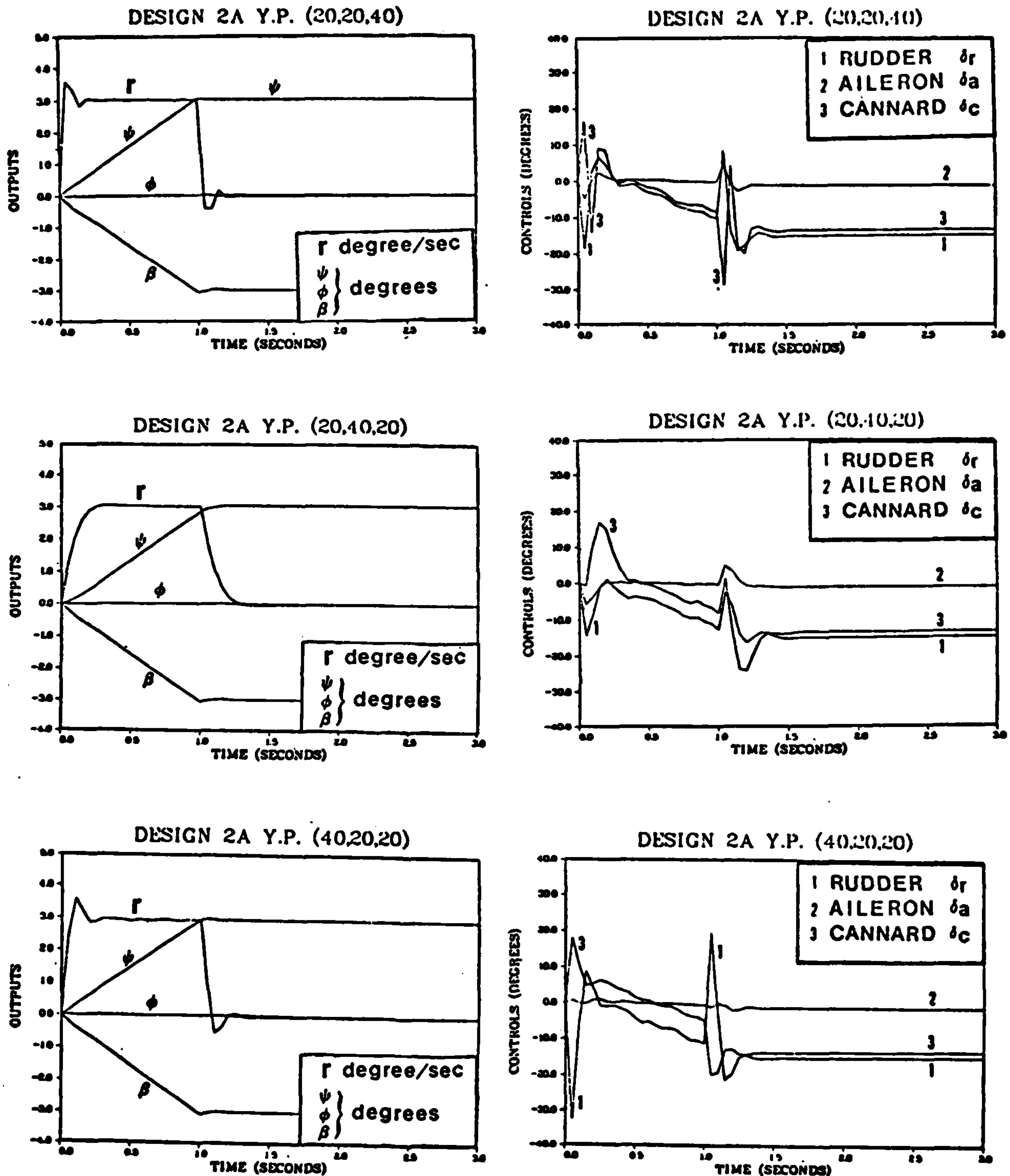


Figure 3-14 One Constant Increased

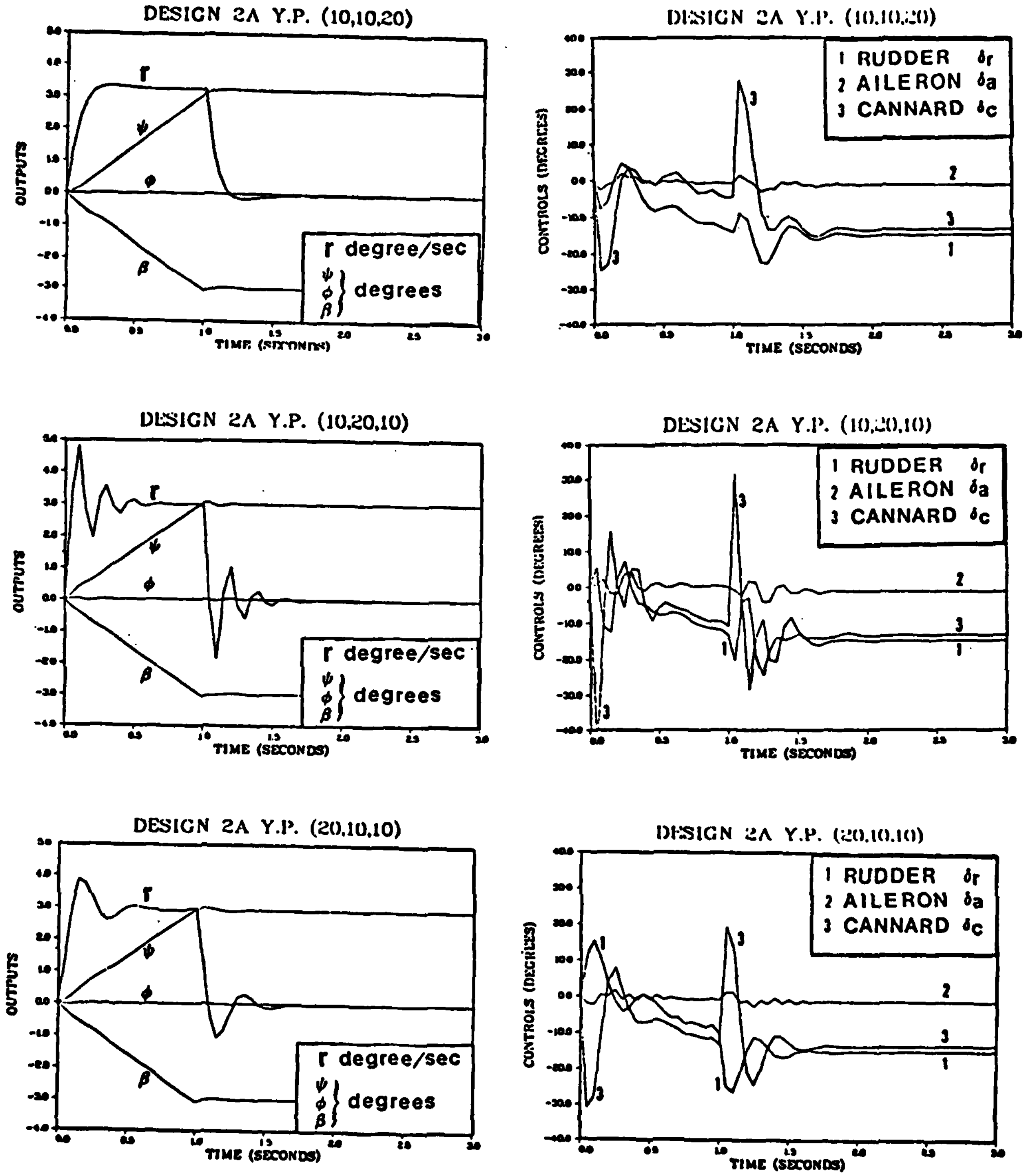


Figure 3-15 Two Constants Decreased

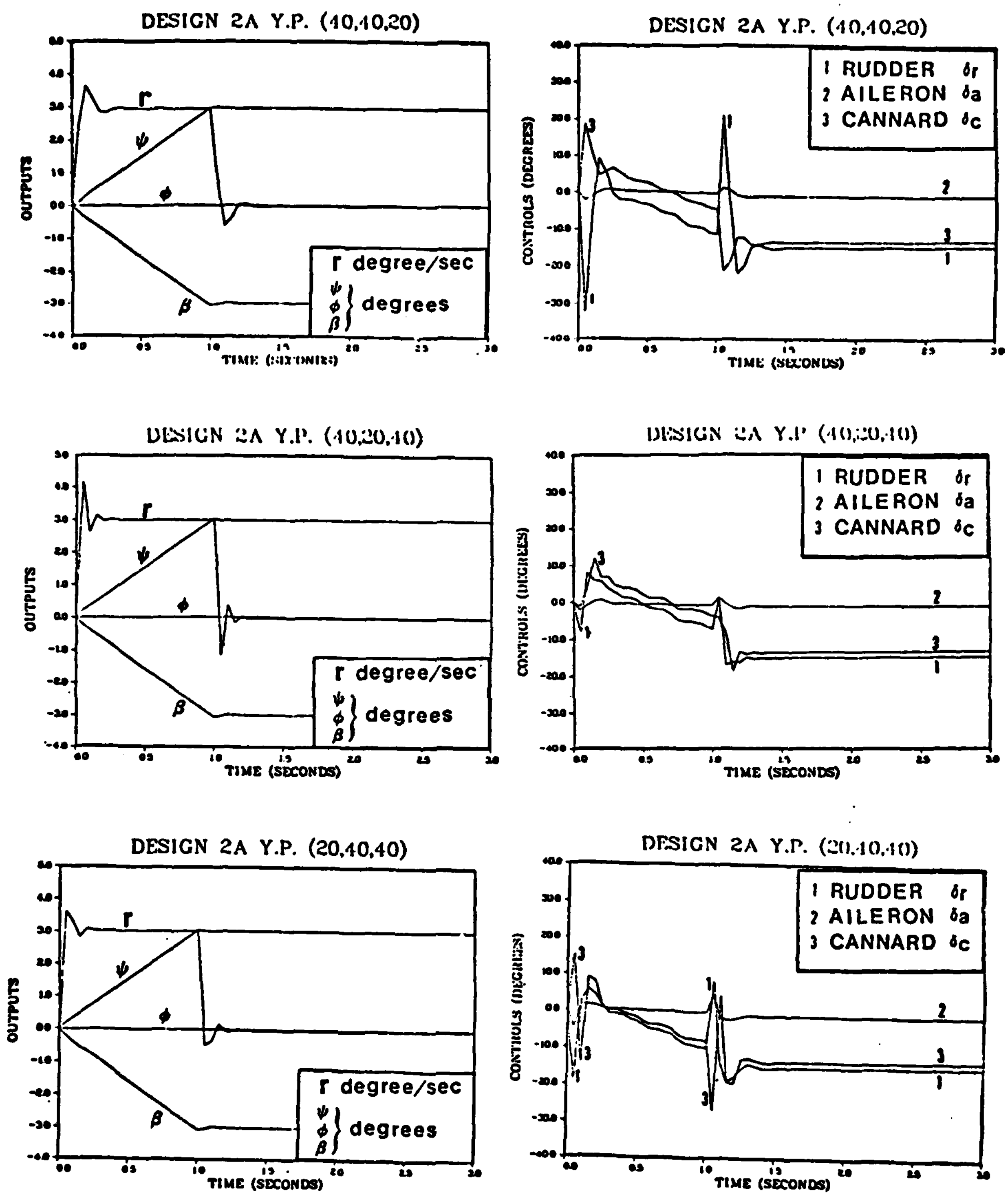


Figure 3-16 Two Constants Increased

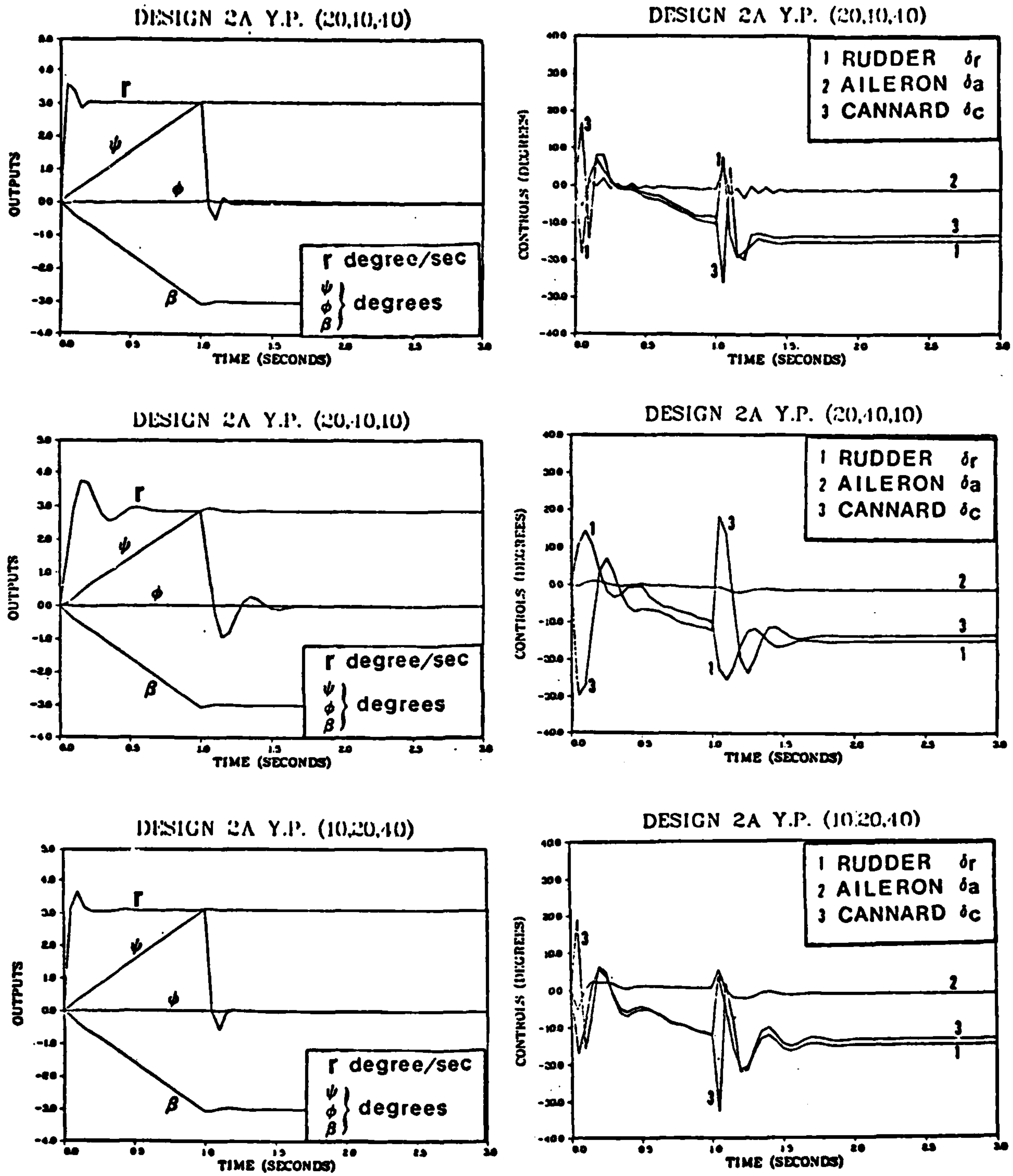
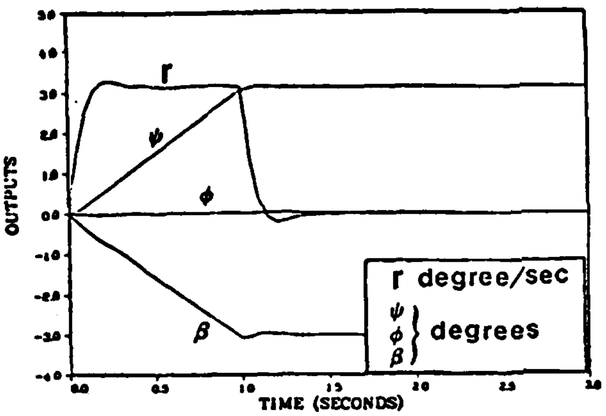
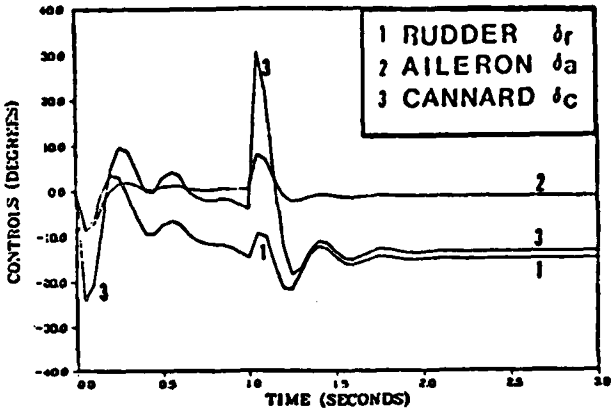


Figure 3-17 One Constant Decreased, One Constant Increased

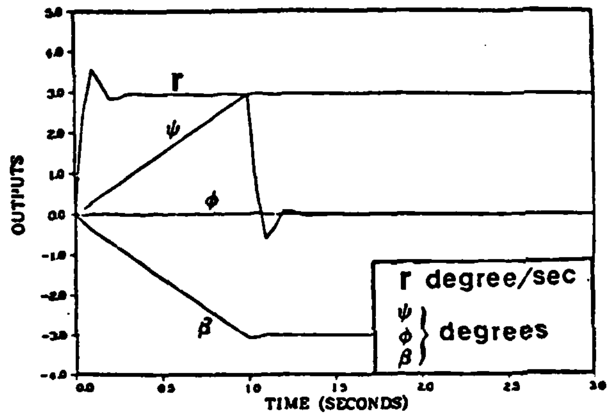
DESIGN 2A Y.P. (10,40,20)



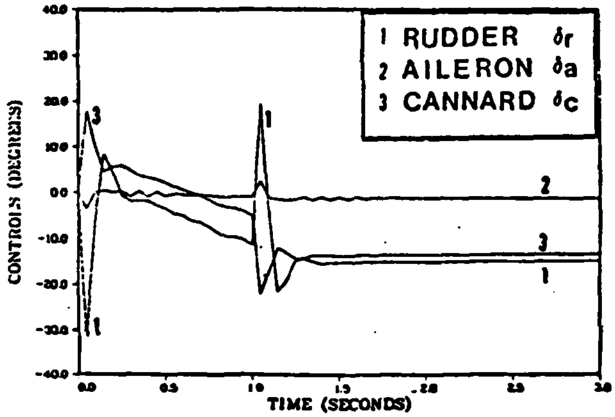
DESIGN 2A Y.P. (10,40,20)



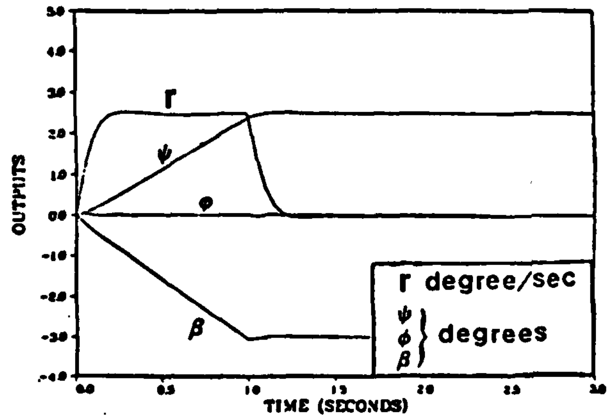
DESIGN 2A Y.P. (40,10,20)



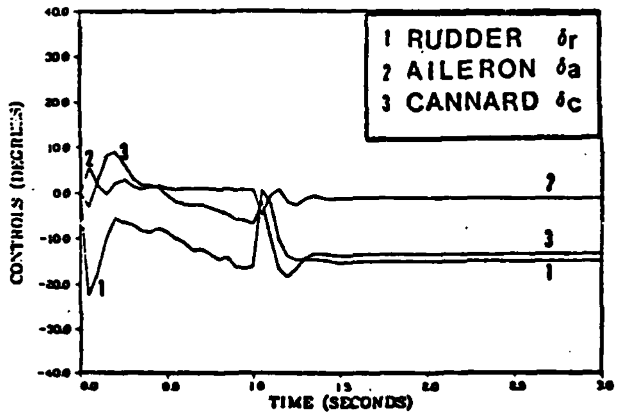
DESIGN 2A Y.P. (40,10,20)



DESIGN 2A Y.P. (40,20,10)



DESIGN 2A Y.P. (40,20,10)



IV. Conclusions and Recommendations

Conclusions

Major conclusions are based on the results of this report. They are first presented in Chapter III and are now summarized.

The design method of Dr Porter is easy to apply and does produce controller designs to perform desired maneuvers. This method is relatively new and promising. Research and applications of this method should continue.

The controllers are robust with respect to varying flight conditions. They are not robust with respect to varying maneuver commands. The controller designed at a medium dynamic pressure flight condition produced the more robust controller design.

Actuator dynamics significantly affect the output and control surface time responses. Simplified models without actuator dynamics included provide misleading results. Actuator dynamics should not be ignored in the design of flight controllers with this method.

Recommendations

This study followed the study performed by Lt Ridgely (Ref 10). His study examines the longitudinal modes of AFTI. This study examines the lateral-directional modes of AFTI. The next study should combine the two modes and examine the response of the total aircraft when performing various maneuvers.

Further robustness examination should be conducted to find if a controller can be designed that is both robust with respect to flight condition and maneuver commands. If so, what parameters affect robustness.

The computer program used for this study is limited when compared with the computer program developed for the design of digital controllers using Porter's method, MULTI (Ref 13). MULTI should be augmented to include the design of analogue controllers.

The precision of program ZERO should be increased from single precision to double precision to increase the program's accuracy. Also the subroutines in ZERO that check the rank of the system matrices should be changed so that complex system zeros can also be tested.

Bibliography

1. Fortmann,T. and Hitz,K. 1977. An Introduction to Linear Control Systems. New York: Marcel Dekker Inc.

2. Kirk,D. 1970. Optimal Control Theory. New Jersey: Prentice-Hall.

3. Laub,A.J. and Moore, B.C. 1977. Calculation of Transmission Zeros Using QZ Techniques. Proceedings of the 1977 IEEE Conference on Decision and Control.

4. MacFarlane,A.G.J. and Karcanias,N. 1976. Poles and Zeros of Linear Multivariable Systems. International Journal of Control. 24 no.1 :33-74.

5. McRuer, Ashkenas, and Graham. 1973. Aircraft Dynamics and Automatic Control. New Jersey: Princeton University Press.

6. Moler, C.B. and Stewart, G.W. 1973. An Algorithm for Generalized Matrix Eigenvalue Problems. SIAM Journal of Numerical Analysis. 24 no.10: 241-255.

7. Porter, B. 1981. Design of High-Performance Tracking Systems. AFWAL/FIGL. Wright-Patterson AFB. Ohio.

8. Porter,B. 1979. Computation of the Zeros of Linear Multivariable Systems. International Journal of System Science. 10 no.12: 1427-1432.

9. Ridgely, Banda, D'Azzo. 1981. Decoupling of High-Gain Multivariable Tracking Systems. AFWAL/FIGCA. Wright-Patterson AFB. Ohio.

10. Ridgely,D. 1981. Use of Entire Eigenstructure Assignment with High-Gain Error-Actuated Flight Control Systems. Masters Thesis. Air Force Institute of Technology. Wright-Patterson AFB. Ohio.

Bibliography

11. Silverthorn,J. and Reid,J.G. 1980. Computation of the Subspaces for Eintire Eigensructure Assignment Via Singular Value Decomposition. Proceedings of the 19th IEEE onference on Decision and Control. 12 no.10: 1206-1207.

12. Wind Tunnel Data, AFTI/F-16. AFWAL/FII. Wright-Patterson AFB. Ohio

13. Porter,D. 1981. Design and Analysis of Multivariable Digital Controler for the A-7D Digital II Aircraft and the Development of an Interactive Computer Design Program. Masters Thesis. Air Force Institute of Technology. Wright-Patterson AFB. Ohio.

Appendix A-1. Development of the Aircraft Equations of Motion

As stated in the introduction, only the lateral-directional, decoupled maneuvers are analyzed in this report. Hence, only the lateral-directional set of equations are developed here. The airframe model is a conventional, small perturbation set of equations, which are linearized about straight and level flight ($\theta_o = \alpha_o$). All four maneuvers analyzed are described by body axis variables, therefore, the equations of motion are body axis equations. The linearized equations in the Laplace domain are (Ref 5):

$$sv + U_o r - w_o p - g \cos \theta_o \phi = Y_v v + Y_p p + Y_r r + \Sigma Y_\delta \delta \tag{A.1}$$

$$sp = L'_v v + L'_p p + L'_r r + \Sigma L'_\delta \delta \tag{A.2}$$

$$sr = N'_v v + N'_p p + N'_r r + \Sigma N'_\delta \delta \tag{A.3}$$

$$s\phi = p + r \tan \theta_o \tag{A.4}$$

These equations are transformed to the time domain, and arranged to have the matrix form

$$\dot{\underline{x}} = \underline{A}\ \underline{x} + \underline{B}\ \underline{u} \tag{A.5}$$

where

$$\underline{x}^T = \left[\phi,\ \beta,\ p,\ r\right] \tag{A.6}$$

$$\underline{u}^T = \left[\delta_r,\ \delta_a,\ \delta_c\right] \tag{A.7}$$

$$\underline{A} = \begin{bmatrix} 0 & , 0, & 1 & , \tan\theta_o \\ g\cos\theta_o/u, & Y_v, & (Y_p + w_o)/u, & (Y_r - u_o)/u \\ 0 & , L'_\beta, & L/_p & , L'_r \\ 0 & , N'_\beta, & N'_p & , N'_r \end{bmatrix} \quad (A.8)$$

$$\underline{B} = \begin{bmatrix} 0 & , 0 & , 0 \\ Y^*_{\delta r} & , Y^*_{\delta a} & , Y^*_{\delta c} \\ L'_{\delta r} & , L'_{\delta a} & , L'_{\delta c} \\ N'_{\delta r} & , N'_{\delta a} & , N'_{\delta c} \end{bmatrix} \quad (A.9)$$

The AFIT-16 data (Ref 12) is not consistent with eqs. A.1 through A.9, since the stability coefficients are in non-dimensional form and are for the stability axis system. Also some of these stability coefficients have units of per radian, while others have units of per degree. The moments of inertia given are in dimensional, body axis form.

The steps used to transform the given data to data consistent with eqs. A.1 through A.9 are:

1. Transform all stability coefficients with units of per degree to per radian.

2. Transform the moments of inertias to the stability axis system.

3. Dimensionalize the stability coefficients.

4. Transform the L and N dimensional stability coefficients to prime form.

5. Transform the dimensional stability coefficients to the body axis sytem.

Ten different flight conditions are used in this study, requiring ten sets of equations of motion. The data for these flight conditions are listed in Table A-1. A computer program is written to perform all of the above transformations. Use of the computer program guarantees that the data transformations are applied consistently. An example and program listing are in Appendix A-2.

The stability coefficients, as originally given, in non-dimensional stability axis form are listed in Table A-2. The dimensional transformation of step 1 consists of multiplication by $180^{o}/\pi$. The transformation of step 2 makes the multiplication factors of steps 3 and 4 consistent, that is, inertias and stability coefficients are all in the stability axis system. The transformation from body axis to stability axis is accomplished by using the following equations:

$$(I_{xx})_s = (I_{xx})_b \cos^2 \theta_o - 2(I_{xz})_b \cos\theta_o \sin\theta_o + (I_{zz})_b \sin^2\theta_o \qquad (A.10)$$

$$(I_{zz})_s = (I_{zz})_b \cos^2 \theta_o + 2(I_{xz})_b \cos\theta_o \sin\theta_o + (I_{xx})_b \sin^2\theta_o \qquad (A.11)$$

$$(I_{xz})_s = ((I_{xx})_b - (I_{zz})_b) \cos\theta_o \sin\theta_o + (I_{xz})_b (\cos^2\theta_o - \sin^2\theta_o) \qquad (A.12)$$

TABLE A-1

Flight Condition Data

Dynamic Pressure, Q (lb/ft^2)	Altitude (feet)	Density, ρ (slug/ft^3)	Sonic Velocity (ft/sec)	Mach	Velocity, U (ft/sec)	Angle of Attack, α_o (degree)
70	30,000	0.000889	994.6	0.4	397.84	11.400
109	20,000	0.001266	1036.8	0.4	414.72	6.952
163	10,000	0.001755	1077.4	0.4	430.96	4.552
223	40,000	0.000585	968.1	0.9	871.29	3.692
436	20,000	0.001266	1036.8	0.8	829.44	2.285
443	5,000	0.002048	1097.1	0.6	658.26	1.824
532	30	0.002377	1116.4	0.6	669.84	1.547
652	10,000	0.001755	1077.4	0.8	861.92	1.733
825	10,000	0.001755	1077.4	0.9	969.66	1.462
1198	30	0.002377	1116.4	0.9	1004.76	1.247

The multiplication factors required to dimensionalize the stability coefficients, step 3, are shown in Table A-3. The transformation of step 4 is accomplished by using the equations

$$L'_i = \frac{L_i + ((I_{xz})_s/(I_{xx})_x)N_i}{(1 - ((I_{xz})_s^2/(I_{xx})_s(I_{zz})_s)} \tag{A.13}$$

$$N'_i = \frac{N_i + ((I_{xz})_s/(I_{zz})_s)L_i}{(1 - ((I_{xz})_s^2/(I_{xx})_s(I_{zz})_s)} \tag{A.14}$$

The fixed aircraft parameters required for steps 3 and 4 for the AFIT-16 are shown in Table A-4. The transformation from the stability axis system to the body axis system, step 5, is accomplished with the following set of equations:

$$(Y_{v,\delta})_b = (Y_{v,\delta})_s \tag{A.15}$$

$$(Y_p)_b = (Y_p)_s \cos\theta_o - (Y_r)_s \sin\theta_o \tag{A.16}$$

$$(Y_r)_b = (Y_r)_s \cos\theta_o + (Y_p)_s \sin\theta_o \tag{A.17}$$

$$(L'_{v,\delta})_b = (L'_{v,\delta})_s \cos\theta_o - (N'_{v,\delta})_s \sin\theta_o \tag{A.18}$$

$$(L'_p)_b = (L'_p)_s \cos^2\theta_o - ((L'_r)_s + (N'_p)_s) \sin\theta_o \cos\theta_o + (N'_p)_s \sin^2\theta_o \tag{A.19}$$

$$(L'_r)_b = (L'_r)_s \cos^2\theta_o - ((N'_r)_s - (L'_p)_s \sin\theta_o \cos\theta_o - (N'_p)_s \sin 2\theta_o \tag{A.20}$$

$$(N'_{v,\delta})_b = (N'_{v,\delta})_s \cos\theta_o + (L'_{v,\delta})_s \sin\theta_o \tag{A.21}$$

$$(N'_p)_b = (N'_p)_s \cos^2 \theta_o - ((N'_r)_s - (L'_p)_s) \sin \theta_o \cos \theta_o$$
$$- (L'_r)_s \sin^2 \theta_o \qquad \text{(A.22)}$$

$$(N'_r)_b = (N'_r)_s \cos^2 \theta_o + ((L'_r)_s + (N'_p)_s) \sin \theta_o \cos \theta_o$$
$$+ (L'_p)_s \sin^2 \theta_o \qquad \text{(A.23)}$$

The side-slip angle, β, is defined as

$$\beta = v/U \qquad \text{(A.24)}$$

The dimensional β stability coefficients differ from the v stability coefficients by a factor of $1/U$. Note, for example, that $N_\beta \beta = N_r v$. The Y^* terms represent a multiplication by $1/U$, that is

$$Y^*_i = Y_i/U \qquad \text{(A.25)}$$

Yhe matrices $\underline{A}$ and $\underline{B}$ for all ten flight conditions are listed in Table A-5.

TABLE A-2

AFTI-16 Non-Dimensional Stability Axis Data

Lateral-Directional CN

Q (lb/ft^2)	C_{n_β} (deg^{-1})	C_{n_p} (rad^{-1})	C_{n_r} (rad^{-1})	$C_{n_{\delta r}}$ (deg^{-1})	$C_{n_{\delta f}}$ (deg^{-1})	$C_{n_{\delta dt}}$ (deg^{-1})	$C_{n_{\delta c}}$ (deg^{-1})
70	.002681	-.038417	-.496924	-.001667	.000379	-.000979	.000934
109	.002121	-.020725	-.492522	-.001618	.000141	-.000986	.001071
163	.001784	-.013125	-.487107	-.001557	.000007	-.000955	.001118
223	.001800	-.010649	-.279610	-.001436	-.000084	-.001069	.001196
436	.001710	-.006850	-.346045	-.001333	-.000144	-.000978	.001215
443	.001775	-.005461	-.476796	-.001379	-.000137	-.000893	.001168
532	.001761	-.004666	-.473181	-.001342	-.000147	-.000862	.001169
652	.001677	-.005165	-.341586	-.001229	-.000149	-.000899	.001217
825	.001636	-.004298	-.273077	-.001137	-.000149	-.000885	.001250
1198	.001447	-.003620	-.260486	-.000947	-.000121	-.000741	.001228

Lateral-Directional CL

Q (lb/ft^2)	C_{l_β} (deg^{-1})	C_{l_p} (rad^{-1})	C_{l_r} (rad^{-1})	$C_{l_{\delta r}}$ (deg^{-1})	$C_{l_{\delta f}}$ (deg^{-1})	$C_{l_{\delta dt}}$ (deg^{-1})	$C_{l_{\delta c}}$ (deg^{-1})
70	-.002583	-.223766	.164036	.000179	-.001508	-.001857	-.000014
109	-.002396	-.240803	.103189	.000323	-.001899	-.001649	.000072
163	-.002162	-.242331	.064776	.000387	-.002032	-.001673	.000127
223	-.002358	-.362413	.087114	.000331	-.002195	-.001836	.000226

TABLE A-2 (cont)

Q (lb/ft^2)	C_{l_β} (deg^{-1})	C_{l_p} (rad^{-1})	C_{l_r} (rad^{-1})	$C_{l_{\delta r}}$ (deg^{-1})	$C_{l_{\delta f}}$ (deg^{-1})	$C_{l_{\delta dt}}$ (deg^{-1})	$C_{l_{\delta c}}$ (deg^{-1})
436	-.001929	-.313529	.031455	.000347	-.001971	-.001810	.000205
443	-.001703	-.232873	.003219	.000392	-.002030	-.001699	.001409
532	-.001608	-.230739	-.004990	.000386	-.001954	-.001656	.000144
652	-.001738	-.306430	.011514	.000326	-.001738	-.001702	.000199
825	-.001742	-.342130	.006361	.000294	-.001523	-.001682	.000224
1198	-.001602	-.328623	-.002132	.000237	-.001184	-.001542	.000217

Lateral-Directional CY

Q (lb/ft^2)	C_{Y_β} (deg^{-1})	C_{Y_p} (rad^{-1})	C_{Y_r} (rad^{-1})	$C_{Y_{\delta r}}$ (deg^{-1})	$C_{Y_{\delta f}}$ (deg^{-1})	$C_{Y_{\delta dt}}$ (deg^{-1})	$C_{Y_{\delta c}}$ (deg^{-1})
70	-.002015	.224381	.603623	.003064	.000703	.002916	.001006
109	-.021784	.189589	.557355	.003069	.000667	.002528	.001075
163	-.020995	.141151	.535556	.003066	.000571	.002132	.001233
223	-.022820	.110272	.556043	.002667	-.000138	.001998	.001350
436	-.021901	.070000	.547367	.002622	-.000113	.001618	.001509
443	-.021095	.058391	.537543	.002991	-.000080	.001364	.000146
532	-.020953	.051413	.537411	.002920	-.000094	.001280	.001462
652	-.021466	.056100	.543785	.002422	-.000112	.001463	.001622
825	-.021582	.045047	.548795	.002110	-.000077	.001492	.001785
1198	-.020906	.038223	.537804	.001745	-.000063	.001208	.001793

TABLE A-3

Lateral-Directional Dimensionalization Factors

$$Y_v = \frac{\rho S U}{2m} C_{y\beta} \qquad N_\beta = \frac{\rho S U^2 b}{2 I_{zz}} C_{n\beta} \qquad L_\beta = \frac{\rho S U^2 b}{2 I_{xx}} C_{l\beta}$$

$$Y_r = \frac{\rho S b}{4m} C_{yr} \qquad N_r = \frac{\rho S U b^2}{4 I_{zz}} C_{nr} \qquad L_r = \frac{\rho S U b^2}{4 I_{xx}} C_{lr}$$

$$Y_\rho = \frac{\rho S b}{4m} C_{y\rho} \qquad N_\rho = \frac{\rho S U b^2}{4 I_{zz}} C_{n\rho} \qquad L_\rho = \frac{\rho S U b^2}{4 I_{xx}} C_{l\rho}$$

$$Y^*_\delta = \frac{\rho S U}{2m} C_{y\delta} \qquad N_\delta = \frac{\rho S U^2 b}{2 I_{zz}} C_{n\delta} \qquad L_\delta = \frac{\rho S U^2 b}{} C_{l\delta}$$

TABLE A-4

AFTI-16 Fixed Aircraft Parameters

weight, w	21018 pounds
span, b	30 feet
Area, S	300 $(\text{feet})^2$
$(I_{xx})_b$	10033.429 slug-ft^2
$(I_{zz})_b$	61278.452 slug-ft^2
$(I_{xz})_b$	282.132 slug-ft^2

TABLE A-5

AFTI-16 Plant Matrices

$$\dot{\bar{x}} = A\bar{x} + B\bar{u}, \quad \bar{x}^T = \left[\phi, \beta, p, r\right] \quad \bar{u}^T = \left[\delta_r, \delta_a, \delta_c\right]$$

Design 1 Q = 109

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .1219 \\ .0770 & -.1506 & .1210 & -.9926 \\ 0.0000 & -14.7042 & -.8989 & .4595 \\ 0.0000 & 1.5960 & .0007 & -.2750 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0212 & .0090 & .0074 \\ 2.8507 & -12.7814 & -.2979 \\ -1.4230 & -.4112 & .9809 \end{bmatrix}$$

Design 2 Q = 443

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0318 \\ .0489 & -.3744 & .0318 & -.9995 \\ 0.0000 & -39.9264 & -2.1137 & .0797 \\ 0.0000 & 6.2376 & -.0063 & -.7074 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0531 & .0046 & .0250 \\ 9.7929 & -55.7388 & 2.6033 \\ -5.0544 & -1.8927 & 4.3880 \end{bmatrix}$$

Design 3 Q = 825

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0255 \\ .0332 & -.4836 & .0255 & -.9997 \\ 0.0000 & -75.3125 & -3.9178 & .0382 \\ 0.0000 & 10.6995 & -.0294 & -.5117 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0473 & .0066 & .0399 \\ 13.4743 & -82.0763 & 8.3887 \\ -7.7774 & -3.2921 & 8.7542 \end{bmatrix}$$

Q = 70

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .2016 \\ .0793 & -.1025 & .1977 & -.9803 \\ 0.0000 & -11.0376 & -.6157 & .4996 \\ 0.0000 & 1.2029 & .0009 & -.1777 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0143 & .0067 & .0047 \\ 1.7994 & -7.0917 & -.7020 \\ -.9382 & -.1855 & .5372 \end{bmatrix}$$

TABLE A-5 (cont)

Q = 163

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0796 \\ .0744 & -.2091 & .0794 & -.9968 \\ 0.0000 & -19.1782 & -1.2617 & .4152 \\ 0.0000 & 2.1153 & -.0009 & -.3992 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0305 & .0110 & .0123 \\ 4.2081 & -20.3236 & .3607 \\ -2.0671 & -.6771 & 1.5439 \end{bmatrix}$$

Q = 223

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0645 \\ .0369 & -.1532 & .0644 & -.9979 \\ 0.0000 & -28.1034 & -1.2593 & .2752 \\ 0.0000 & 2.9444 & -.0149 & -.1532 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0179 & .0024 & .0091 \\ 4.7528 & -30.0084 & 1.7593 \\ -2.6169 & -1.1128 & 2.2662 \end{bmatrix}$$

Q = 436

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0399 \\ .0388 & -.3029 & .0399 & -.9992 \\ 0.0000 & -44.5171 & -2.2233 & .2199 \\ 0.0000 & 5.7767 & -.0167 & -.3982 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0363 & .0040 & .0209 \\ 8.8177 & -53.9258 & 3.6280 \\ -4.7193 & -2.0255 & 4.4971 \end{bmatrix}$$

Q = 532

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0270 \\ .0480 & -.4392 & .0270 & -.9996 \\ 0.0000 & -45.1440 & -2.4708 & -.0066 \\ 0.0000 & 7.4966 & -.0081 & -.8300 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0612 & .0047 & .0306 \\ 11.4009 & -64.6667 & 3.2286 \\ -5.9204 & -3.2106 & 5.2760 \end{bmatrix}$$

TABLE A-5 (cont)

$Q = 652$

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0303 \\ .0373 & -.4275 & .0302 & -.9995 \\ 0.0000 & -59.6596 & -3.1213 & .1119 \\ 0.0000 & 8.6325 & -.0212 & -.5683 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0482 & .0051 & .0323 \\ 11.9760 & -72.1512 & 5.6206 \\ -6.6298 & -.7405 & 6.7320 \end{bmatrix}$$

$Q = 1198$

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & .0218 \\ .0320 & -.6574 & .0218 & -.9998 \\ 0.0000 & -100.3136 & -5.2784 & -.0773 \\ 0.0000 & 13.7921 & -.0377 & -.6860 \end{bmatrix} \quad B = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ .0549 & .0075 & .0564 \\ 15.6158 & -96.4546 & 12.0815 \\ -9.4350 & -3.880 & 12.4986 \end{bmatrix}.$$

Appendix A-2

Dimensionalization Computer Program

Example and Program Listing

The example shown is that of Design 2.

$$Q = 443 \text{ lb/ft}^2.$$

```
 THIS PROGRAM CONVERTS NON-DIMENSIONAL-STABILITY AXIS-
 DATA TO DIMENSIONAL-BODY AXIS-DATA FOR THE AFTI-16

 SELECT MODE.  LONGITUDNAL ONLY (1)
               LATERAL ONLY (2)
               BOTH (3)  : 2
STANDARD AFTI AIRCRAFT?
 YES=1.   NO=0.  : 1

 FIXED AIRCRAFT PARAMETERS

 WEIGHT (LBS) :21018.
 SPAN (FT)    :30.
 AREA (FT **2):300.
 MAC (FT)     :11.32
 I-X          :10033.429
 I-Y          :53876.269
 I-Z          :61278.452
 I-XZ         :282.13217


 INPUT VARING PARMETERS

 HEIGHT (FT) : 5000
 MACH : 0.6
 SONIC VELOCITY (FT/SEC) : 1097.1
 DENSITY (SLUG/FT **3) : 0.002048
 ANGLE OF ATTACK (DEG) : 1.824

 MOMENT COEF.

 CL : .1572
 CM : .0026
 CD : .0231

 INPUT THE LATERAL STABILITY COEFICIENTS

            CN , CL , CY
 BETA (DEG) : .001775,-.001703,-.021095
 P (RAD) : -.005461,-.232873,.058391
 R (RAD) : -.476796,.003219,.537543
 RUDDER (DEG) : -.001379,.000392,.002991
 FLAPERON (DEG) : -.000137,-.002030,-.000080
 DIFFAIL (DEG) : -.000893,-.001699,.001364
 CANNARD (DEG) : .001168,.000146,.001409
  INPUT ALERON COMBINATION TERMS
  ALERON= A*FLAPERON + B*DIFF TAIL
   ENTER A, B : 1,.25
```

```
HEIGHT=5000. (FT)     MACH=.6

 BODY AXIS DATA -LATERAL-

                 N          L          Y

   BETA       6.23792738   -39.92845659     -.37444266   (VEL.)

    P         -.00631482    -2.11380615      .00029122

    R         -.70740908     .07970246       .00380601

 RUDDER     -5.05463826    9.79342839      .05309116

 AILERONS     -1.89279772   -55.74144715     .00463283

 CANNARD      4.38821795    2.60330899      .02501018

THE A-MATRIX IS :

  -.37444266      .03182987    -.99948753     .04885258
-39.92845659    -2.11380615     .07970246    0.00000000
  6.23792738     -.00631482    -.70740908    0.00000000
  0.00000000     1.00000000     .03184556    0.00000000

 THE STATE VECTOR TRANSPOSE IS :
 ( BETA , P , R , PHI )

 THE B-MATRIX IS :

  .05309116      .00463283      .02501018
 9.79342839   -55.74144715     2.60330899
-5.05463826    -1.89279772     4.38821795
 0.00000000     0.00000000     0.00000000

 THE CONTROL VECTOR TRANSPOSE IS:
 (RUDDER, AILERON, CANNARD)

 NEW FLIGHT CONDITON?
 YES=1.  NO=0.  : 0
   STOP
   024600 MAXIMUM EXECUTION FL.
    0.103 CP SECONDS EXECUTION TIME.
COMMAND-
```

```
      PROGRAM DIMEN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      REAL B1(8),B2(8),C1(9),C2(9),C3(9),D1(9),D2(9),D3(9)
      REAL E1(9),E2(9),E3(9),F1(9),F2(9),F3(9),G1(2),G2(2)
      REAL C4(9),C5(9),C6(9),D4(9),D5(9),D6(9),F4(9),F5(9),F6(9)
      REAL G4(9),G5(9),G6(9),AMTL(4,4),BMTL(4,3)
      REAL G3(2),AMAT(4,4),BMAT(4,3)
C
      PRINT*," "
      PRINT*,"    THIS PROGRAM CONVERTS NON-DIMENSIONAL-STABILITY AXIS-"
      PRINT*,"    DATA TO DIMENSIONAL-BODY AXIS-DATA FOR THE AFTI-16"
  500 PRINT*," "
      PRINT*,"    SELECT MODE.  LONGITUDNAL ONLY (1)"
      PRINT*,"                  LATERAL ONLY (2)"
      PRINT*,"                  BOTH (3)  :"
      READ*,I6
C
C        THE FIXED AIRCRAFT PARAMETERS ARE CONTAINED IN THE
C        VECTOR B1.
C
      PRINT*,"  STANDARD AFTI AIRCRAFT?"
      PRINT*,"   YES=1.   NO=0.  :"
      READ*,I1
      CALL AFTI(B1,I1)
C
      PRINT*," "
      PRINT*,"   FIXED AIRCRAFT PARAMETERS"
      PRINT*," "
      PRINT*,"   WEIGHT (LBS) :",B1(1)
      PRINT*,"   SPAN (FT)    :",B1(2)
      PRINT*,"   AREA (FT **2):",B1(3)
      PRINT*,"   MAC (FT)     :",B1(4)
      PRINT*,"   I-X          :",B1(5)
      PRINT*,"   I-Y          :",B1(6)
      PRINT*,"   I-Z          :",B1(7)
      PRINT*,"   I-XZ         :",B1(8)
      PRINT*," "
C
C
C       THE PARAMETERS THAT VARY WITH FLIGHT CONDITIONS ARE
C       CONTAINED IN THE VECTOR B2
C
      PRINT*," "
      PRINT*,"   INPUT VARING PARMETERS"
      PRINT*," "
      PRINT*,"   HEIGHT (FT) :"
      READ*,B2(1)
      PRINT*,"   MACH :"
      READ*,B2(2)
      PRINT*,"   SONIC VELOCITY (FT/SEC) :"
      READ*,B2(3)
      PRINT*,"   DENSITY (SLUG/FT **3) :"
      READ*,B2(4)
      PRINT*,"   ANGLE OF ATTACK (DEG) :"
      READ*,B2(5)
```

```
      PRINT*," "
      PRINT*,"   MOMENT COEF."
      PRINT*," "
      PRINT*,"   CL :"
      READ*,B2(6)
      PRINT*,"   CM :"
      READ*,B2(7)
      PRINT*,"   CD :"
      READ*,B2(8)
C
C         THE NON DIMENSIONAL-STABILITY AXIS-DATA ARE CONTAINED
C         IN THE VECTORS C1, C2, C3, C4, C5, C6.
C
C           LATERAL-DIRECTIONAL
C
C         C1 CONTAINS THE N-MOMENT COEFICIENTS
C         C2 CONTAINS THE L-MOMENT COEFICIENTS
C         C3 CONTAINS THE Y-FORCE COEFICIENTS
C
C           LONGITUDINAL
C
C         C4 CONTAINS THE D-FORCE COEFICIENTS
C         C5 CONTAINS THE M-MOMENT COEFICEINTS
C         C6 CONTAINS THE L-FORCE COEFICEINTS
C
      IF(I6.EQ.1) GO TO 1200
C
      PRINT*," "
      PRINT*,"   INPUT THE LATERAL STABILITY COEFICIENTS"
      PRINT*," "
      PRINT*,"                CN , CL , CY"
      PRINT*,"   BETA (DEG) :"
      READ*,C1(1),C2(1),C3(1)
      PRINT*,"   P (RAD) :"
      READ*,C1(2),C2(2),C3(2)
      PRINT*,"   R (RAD) :"
      READ*,C1(3),C2(3),C3(3)
      PRINT*,"   RUDDER (DEG) :"
      READ*,C1(4),C2(4),C3(4)
      PRINT*,"   FLAPERON (DEG) :"
      READ*,C15,C25,C35
      PRINT*,"   DIFFAIL (DEG) :"
      READ*,C15A,C25A,C35A
      PRINT*,"   CANNARD (DEG) :"
      READ*,C1(6),C2(6),C3(6)
      PRINT*,"    INPUT ALERON COMBINATION TERMS"
      PRINT*,"    ALERON= A*FLAPERON + B*DIFF TAIL"
      PRINT*,"     ENTER A, B :"
      READ*,X7,X8
      C1(5)=(X7*C15)+(X8*C15A)
      C2(5)=(X7*C25)+(X8*C25A)
      C3(5)=(X7*C35)+(X8*C35A)
C
C
      IF(I6.EQ.2) GO TO 1300
```

```
C
 1200 PRINT*," "
      PRINT*,"    INPUT THE LONGITUDINAL STABILITY COEFICEINTS"
      PRINT*," "
      PRINT*,"                     CD     CM     CL"
      PRINT*," "
      PRINT*,"    VELOCITY (FT/SEC) :"
      READ*,C4(1),C5(1),C6(1)
      PRINT*,"    ALPHA (DEG) :"
      READ*,C4(2),C5(2),C6(2)
      PRINT*,"    ELEVATOR (DEG) :"
      READ*,C4(3),C5(3),C6(3)
      PRINT*,"    T.E. FLAP (DEG) :"
      READ*,C4(4),C5(4),C6(4)
      PRINT*,"    L.E. FLAP (DEG) :"
      READ*,C4(5),C5(5),C6(5)
      PRINT*,"    Q (RAD) ** CM AND CL ONLY ** :"
      READ*,C5(6),C6(6)
      PRINT*,"    ALPHA DOT (RAD) ** CM AND CL ONLY ** :"
      READ*,C5(7),C6(7)
C
C        CONVERT B2(5) ANGLE OF ATTACK FROM
C        DEGREES TO RADIANS.
C
 1300 PI=3.141592654
      DTR=180.0/PI
C
      B2(5)=B2(5)/DTR
C
      CA=COS(B2(5))
      SA=SIN(B2(5))
C
C      CONVERT MOMENTS OF INERTIA FROM BODY AXIS TO
C      STABILITY AXIS TO MAKE FUTURE CALCULATIONS CONSISTANT.
C
      B15=(B1(5)*(CA**2))-(2.0*B1(8)*CA*SA)+(B1(7)*(SA**2))
      B17=(B1(7)*(CA**2))+(2.0*B1(8)*CA*SA)+(B1(5)*(SA**2))
      B18=((B1(5)-B1(7))*CA*SA)+(B1(8)*((CA**2)-(SA**2)))
C
      B1(5)=B15
      B1(7)=B17
      B1(8)=B18
C
C        CONVERT B1(1) -WEIGHT(LBS)- TO
C                      -MASS(LBS*(SEC**2)/FT)-
C
      B1(1)=B1(1)/32.2
C
C      CALCULATE VELOCITY  -V- =B2(2)*B2(3)
C            B2(2) -MACH-
C            B2(3) -SONIC VELOCITY-
C
      V=B2(2)*B2(3)
C
      U0=V*CA
```

```
      WO=V*SA
C
C        CONVERT STABILITY DERIVATIVES WITH (1/DEG) TO (1/RAD)
C        SPECIFICALLY BETA,RUDDER,AILERONS,AND CANNARDS LATERAL
C          ALPHA,ELEVATOR, AND FLAPS LONGITUDNAL
C
C
      IF(I6.EQ.1) GO TO 1210
C
      C1(1)=C1(1)*DTR
      C2(1)=C2(1)*DTR
      C3(1)=C3(1)*DTR
      C1(4)=C1(4)*DTR
      C2(4)=C2(4)*DTR
      C3(4)=C3(4)*DTR
      C1(5)=C1(5)*DTR
      C2(5)=C2(5)*DTR
      C3(5)=C3(5)*DTR
      C1(6)=C1(6)*DTR
      C2(6)=C2(6)*DTR
      C3(6)=C3(6)*DTR
C
      IF(I6.EQ.2) GO TO 1310
C
 1210 C4(2)=C4(2)*DTR
      C5(2)=C5(2)*DTR
      C6(2)=C6(2)*DTR
      C4(3)=C4(3)*DTR
      C5(3)=C5(3)*DTR
      C6(3)=C6(3)*DTR
      C4(4)=C4(4)*DTR
      C5(4)=C5(4)*DTR
      C6(4)=C6(4)*DTR
      C4(5)=C4(5)*DTR
      C5(5)=C5(5)*DTR
      C6(5)=C6(5)*DTR
C
C
C      G1 G2 G3 CONTAIN THE CONVERSION
C      FACTORS TO CONVERT NON-DIMEN DATA
C      TO DIMEN DATA IN THE STABILITY AXIS
C
 1310 IF(I6.EQ.1) GO TO 1220
C
      G1(1)=(B2(4)*B1(3)*(V**2)*B1(2))/(2.*B1(7))
      G1(2)=(B2(4)*B1(3)*V*(B1(2)**2))/(4.*B1(7))
      G2(1)=G1(1)*B1(7)/B1(5)
      G2(2)=G1(2)*B1(7)/B1(5)
      G3(1)=(B2(4)*B1(3)*V)/(2.*B1(1))
      G3(2)=(B2(4)*B1(3)*B1(2))/(4.*B1(1))
      CALL DIMN(C1,G1,D1)
      CALL DIMN(C2,G2,D2)
      CALL DIMN(C3,G3,D3)
C
C      THE VECTORS D1, D2, D3 CONTAIN THE
```

```
C      DIMEN DATA IN THE STABILITY AXIS
C
      CALL PRM(D1,D2,E1,E2,B1)
C
C      THE VECTORS E1,E2 CONTAIN THE DIMEN
C      DATA IN THE  STABILITY N AXIS EXCEPT
C      THE DATA IS PRIMED,( SEE MC RUER, AIRCRAFT
C      DYNAMICS AND AUTOMATIC CONTROL, PG 257)
C
C        THE DIMEN DATA -BODY AXIS- ARE CONTAINED IN
C        THE VECTORS F1,F2,F3
C
      F1(1)=(E1(1)*CA)+(E2(1)*SA)
      F1(2)=(E1(2)*(CA**2))-((E1(3)-E2(2))*CA*SA)-(E2(3)*(SA**2))
      F1(3)=(E1(3)*(CA**2))+((E2(3)+E1(2))*CA*SA)+(E2(2)*(SA**2))
      DO 10 I=4,6
   10 F1(I)=(E1(I)*CA)+(E2(I)*SA)
C
      F2(1)=(E2(1)*CA)-(E1(1)*SA)
      F2(2)=(E2(2)*(CA**2))-((E2(3)+E1(2))*CA*SA)+(E1(3)*(SA**2))
      F2(3)=(E2(3)*(CA**2))-((E1(3)-E2(2))*CA*SA)-(E1(2)*(SA**2))
      DO 20 I=4,6
   20 F2(I)=(E2(I)*CA)-(E1(I)*SA)
C
      F3(1)=D3(1)
      F3(2)=(D3(2)*CA)-(D3(3)*SA)
      F3(3)=(D3(3)*CA)+(D3(2)*SA)
      DO 30 I=4,6
   30 F3(I)=D3(I)
C
      PRINT*," "
      PRINT*,"   HEIGHT=",B2(1)," (FT)     MACH=",B2(2)
      PRINT*," "
      PRINT*,"    BODY AXIS DATA -LATERAL-"
      PRINT*," "
      PRINT*,"                N        L         Y"
      PRINT*," "
      PRINT 50,F1(1),F2(1),F3(1)
      PRINT 51,F1(2),F2(2),F3(2)
      PRINT 52,F1(3),F2(3),F3(3)
      PRINT 53,F1(4),F2(4),F3(4)
      PRINT 54,F1(5),F2(5),F3(5)
      PRINT 55,F1(6),F2(6),F3(6)
   50 FORMAT(5X,"  BETA ",5X,3(F16.9,3X)," (VEL.)",/)
   51 FORMAT(5X,"   P   ",5X,3(F16.9,3X),/)
   52 FORMAT(5X,"   R   ",5X,3(F16.9,3X),/)
   53 FORMAT(5X,"RUDDER",5X,3(F16.9,3X),/)
   54 FORMAT(5X,"ALERONS",5X,3(F16.9,3X),/)
   55 FORMAT(5X,"CANNARD",5X,3(F16.9,3X))
C
C        PUT EQUATIONS OF MOTION IN MATRIX FORM.
C        SEE MCRUER PAGE 259
C
      DO 60 I=1,4
      DO 60 J=1,4
```

```
   60 AMAT(I,J)=0.0
      DO 70 I=1,4
      DO 70 J=1,3
   70 BMAT(I,J)=0.0
C
      U0=V*CA
      W0=V*SA
C
      AMAT(1,1)=F3(1)
      AMAT(1,2)=(F3(2)+W0)/V
      AMAT(1,3)=(F3(3)-U0)/V
      AMAT(1,4)=32.174*CA/V
      AMAT(2,1)=F2(1)
      AMAT(2,2)=F2(2)
      AMAT(2,3)=F2(3)
      AMAT(3,1)=F1(1)
      AMAT(3,2)=F1(2)
      AMAT(3,3)=F1(3)
      AMAT(4,2)=1.0
      AMAT(4,3)=SA/CA
      BMAT(1,1)=F3(4)
      BMAT(1,2)=F3(5)
      BMAT(1,3)=F3(6)
      BMAT(2,1)=F2(4)
      BMAT(2,2)=F2(5)
      BMAT(2,3)=F2(6)
      BMAT(3,1)=F1(4)
      BMAT(3,2)=F1(5)
      BMAT(3,3)=F1(6)
C
      PRINT*," "
      PRINT*,"  THE A-MATRIX IS :"
      CALL PRINTM(4,4,AMAT)
      PRINT*," "
      PRINT*,"   THE STATE VECTOR TRANSPOSE IS :"
      PRINT*,"   ( BETA , P , R , PHI )"
      PRINT*," "
      PRINT*,"   THE B-MATRIX IS :"
      CALL PRINTM(4,3,BMAT)
      PRINT*," "
      PRINT*,"   THE CONTROL VECTOR TRANSPOSE IS:"
      PRINT*,"   (RUDDER, ALERON, CANNARD)"
C
      IF(I6.EQ.2) GOTO 1320
C
C      THE VECTORS G4,G5,G6 CONTAIN THE FACTORS
C      THAT CONVERT NON DIMENSIONAL DATA TO
C      DIMENSIONAL DATA -STABILITY AXIS-
C
 1220 G4(1)=B2(4)*B1(3)*V/B1(1)
      G4(2)=G4(1)/2.
      G4(3)=B2(4)*B1(3)*(V**2)/(2.*B1(1))
C
      G5(1)=B2(4)*B1(3)*V*B1(4)/B1(6)
      G5(2)=G5(1)/2.
```

```
      G5(3)=B2(4)*B1(3)*(B1(4)**2)/(4.*B1(6))
      G5(4)=G5(3)*V
      G5(5)=B2(4)*B1(3)*(V**2)*B1(4)/(2.*B1(6))
C
      G6(1)=G4(1)
      G6(2)=G4(2)
      G6(3)=B2(4)*B1(3)*B1(4)/(4.*B1(1))
      G6(4)=G6(3)*V
      G6(5)=B2(4)*B1(3)*(V**2)/(2.*B1(1))
C
C       THE VECTORS D4,D5,D6 CONTAIN THE DIMENSIONAL
C       DATA -STABILITY AXIS-
C
C        ** NOTE THE VECTOR SEQUENCE CHANGE **
C
C        1 -- VELOCITY --
C        2 -- W --
C        3 -- W DOT --
C        4 -- Q --
C        5 -- ELEVATOR --
C        6 -- T.E. FLAP -
C        7 -- L.E. FLAP --
C
      D4(1)=G4(1)*(-B2(8)-C4(1))
      D4(2)=G4(2)*(B2(6)-C4(2))
      D4(3)=0.0
      D4(4)=0.0
C
      DO 700 I=5,7
  700 D4(I)=G4(3)*(-C4(I-2))
C
      D5(1)=G5(1)*(B2(7)+C5(1))
      D5(2)=G5(2)*C5(2)
      D5(3)=G5(3)*C5(7)
      D5(4)=G5(4)*C5(6)
C
      DO 710 I=5,7
  710 D5(I)=G5(5)*C5(I-2)
C
      D6(1)=G6(1)*(-B2(6)-C6(1))
      D6(2)=G6(2)*(-C6(2)-B2(8))
      D6(3)=G6(3)*C6(7)
      D6(4)=G6(4)*(-C6(6))
C
      DO 720 I=5,7
  720 D6(I)=G6(5)*(-C6(I-2))
C
C      THE VECTORS F4,F5,F6 CONTAIN THE
C      DATA IN BODY AXIS
C
      F4(1)=(D4(1)*(CA**2))-((D4(2)+D6(1))*SA*CA)+(D6(2)*(SA**2))
      F4(2)=(D4(2)*(CA**2))+((D4(1)-D6(2))*SA*CA)-(D6(1)*(SA**2))
      F4(3)=(D4(3)*(CA**2))-(D6(3)*SA*CA)
C
      DO 800 I=4,7
```

```
  800 F4(I)=(D4(I)*CA)-(D6(I)*SA)
C
      F5(1)=(D5(1)*CA)-(D5(2)*SA)
      F5(2)=(D5(2)*CA)+(D5(1)*SA)
      F5(3)=D5(3)*CA
C
      DO 810 I=4,7
  810 F5(I)=D5(I)
C
      F6(1)=(D6(1)*(CA**2))-((D6(2)-D4(1))*SA*CA)-(D4(2)*(SA**2))
      F6(2)=(D6(2)*(CA**2))+((D6(1)+D4(2))*SA*CA)+(D4(1)*(SA**2))
      F6(3)=(D6(3)*(CA**2))+(D4(3)*SA*CA)
C
      DO 820 I=4,7
  820 F6(I)=(D6(I)*CA)+(D4(I)*SA)
C
      PRINT*," "
      PRINT*,"    HEIGHT :",B2(1),"  MACH :",B2(2)
      PRINT*," "
      PRINT*,"    BODY AXIS DATA -LONGITUDINAL-"
      PRINT*," "
      PRINT*,"                X         M         Z"
      PRINT*," "
      PRINT 90,F4(1),F5(1),F6(1)
      PRINT 91,F4(2),F5(2),F6(2)
      PRINT 92,F4(3),F5(3),F6(3)
      PRINT 93,F4(4),F5(4),F6(4)
      PRINT 94,F4(5),F5(5),F6(5)
      PRINT 95,F4(6),F5(6),F6(6)
      PRINT 96,F4(7),F5(7),F6(7)
C
   90 FORMAT(5X,"VELOCITY",3X,3(F16.9,3X),/)
   91 FORMAT(5X,"   W    ",3X,3(F16.9,3X),/)
   92 FORMAT(5X," W-DOT  ",3X,3(F16.9,3X),/)
   93 FORMAT(5X,"   Q    ",3X,3(F16.9,3X),/)
   94 FORMAT(5X,"ELEVATOR",3X,3(F16.9,3X),/)
   95 FORMAT(5X,"T.E.FLAP",3X,3(F16.9,3X),/)
   96 FORMAT(5X,"L.E.FLAP",3X,3(F16.9,3X),/)
C
      Z3=1.0/(1.0-F6(3))
C
      AMTL(1,1)=F4(1)+(F4(3)*F6(1)*F6(3))
      AMTL(1,2)=(F4(2)*U0)+(F4(3)*U0*F6(2)*Z3)
      AMTL(1,3)=(F4(4)-W0)+(F4(3)*(F6(4)+U0)*Z3)
      AMTL(1,4)=-32.174*CA*(1.0-(F4(3)*Z3))
C
      AMTL(2,1)=F6(1)*Z3/U0
      AMTL(2,2)=F6(2)*Z3
      AMTL(2,3)=(F6(4)+U0)*Z3/U0
      AMTL(2,4)=-32.174*CA*Z3/U0
C
      AMTL(3,1)=F5(1)+(F5(3)*F6(1)*Z3)
      AMTL(3,2)=(F5(2)*U0)+(F5(3)*U0*F6(2)*Z3)
      AMTL(3,3)=F5(4)+(F5(3)*(F6(4)+U0)*Z3)
      AMTL(3,4)=-32.174*CA*F5(3)*Z3
C
```

```
      AMTL(4,1)=0.0
      AMTL(4,2)=0.0
      AMTL(4,3)=1.0
      AMTL(4,4)=0.0
C
      BMTL(1,1)=F4(5)+(F4(3)*F6(5)*Z3)
      BMTL(1,2)=F4(6)+(F4(3)*F6(6)*Z3)
      BMTL(1,3)=F4(7)+(F4(3)*F6(7)*Z3)
C
      BMTL(2,1)=F6(5)*Z3/U0
      BMTL(2,2)=F6(6)*Z3/U0
      BMTL(2,3)=F6(7)*Z3/U0
C
      BMTL(3,1)=F5(5)+(F5(3)*F6(5)*Z3)
      BMTL(3,2)=F5(6)+(F5(3)*F6(6)*Z3)
      BMTL(3,3)=F5(7)+(F5(3)*F6(7)*Z3)
C
      BMTL(4,1)=0.0
      BMTL(4,2)=0.0
      BMTL(4,3)=0.0
C
C
 1320 PRINT*," "
      PRINT*,"   NEW FLIGHT CONDITON?"
      PRINT*,"   YES=1.  NO=0.  :"
      READ*,I2
      IF(I2.EQ.1) GO TO 500
C
C
C
      STOP
      END
C
      SUBROUTINE DIMN(A1,G1,A2)
      REAL A1(9),G1(2),A2(9)
      A2(1)=A1(1)*G1(1)
      DO 10 J=2,3
   10 A2(J)=A1(J)*G1(2)
      DO 20 K=4,6
   20 A2(K)=A1(K)*G1(1)
      RETURN
      END
C
      SUBROUTINE PRM(A1,A2,E1,E2,B1)
      REAL A1(9),A2(9),E1(9),E2(9),B1(8)
      P1=1-((B1(8)**2)/(B1(7)*B1(5)))
      P2=B1(8)/B1(5)
      P3=B1(8)/B1(7)
      DO 10 I=1,6
      E1(I)=(A1(I)+(P3*A2(I)))/P1
   10 E2(I)=(A2(I)+(P2*A1(I)))/P1
      RETURN
      END
C
      SUBROUTINE AFTI(B1,I1)
```

```
      REAL B1(7)
      IF(I1.EQ.1) GO TO 100
C
      PRINT*," "
      PRINT*,"   INPUT FIXED AIRCRAFT PARAMETERS :"
      PRINT*," "
      PRINT*,"   WEIGHT (LBS) :"
      READ*,B1(1)
      PRINT*," "
      PRINT*,"   SPAN (FT) :"
      READ*,B1(2)
      PRINT*," "
      PRINT*,"   AREA (FT **2) :"
      READ*,B1(3)
      PRINT*," "
      PRINT*,"   MEAN AERODYNAMIC CHORD (FT) :"
      READ*,B1(4)
      PRINT*," "
      PRINT*,"   MOMENTS OF INERTIA "
      PRINT*," "
      PRINT*,"   I-X :"
      READ*,B1(5)
      PRINT*," "
      PRINT*,"   I-Y :"
      READ*,B1(6)
      PRINT*," "
      PRINT*,"   I-Z :"
      READ*,B1(7)
      PRINT*," "
      PRINT*,"   I-XZ :"
      READ*,B1(8)
      GO TO 200
  100 B1(1)=21018.0
      B1(2)=30.0
      B1(3)=300.0
      B1(4)=11.32
      B1(5)=10033.429
      B1(6)=53876.269
      B1(7)=61278.452
      B1(8)=282.13217
  200 RETURN
      END
C
      SUBROUTINE PRINTM(N,L,A)
      REAL A(4,4)
      PRINT*," "
C
      DO 100 I=1,N
      PRINT 200,(A(I,J),J=1,L)
  100 CONTINUE
C
  200 FORMAT(1X,8(F15.8,3X))
      RETURN
      END
*EOR
*EOF
```

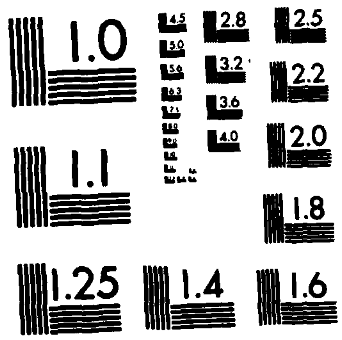
AD-A124 871 HIGH-GAIN ERROR ACTUATED FLIGHT CONTROL SYSTEMS FOR CONTINUOUS LINEAR MUL..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. T LEWIS 2/2

UNCLASSIFIED DEC 82 AFIT/GAE/EE/82D-1 F/G 1/3 NL

END

FILMED

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Appendix B-1

Computer Program to Calculate System Zeros

As part of this thesis effort, Program ZERO is developed to calculate the zeros of the linear matrix system

$$\begin{aligned} \dot{\underline{x}} &= \underline{A}\,\underline{x} + \underline{B}\,\underline{u} \\ \dot{\underline{y}} &= \underline{C}\,\underline{x} + D\,\underline{u} \end{aligned} \tag{B.1}$$

ZERO calculates transmission and decoupling zeros by solving the appropriate generalized eigenvalue problem with the powerful QZ technique (Ref 6). Three specific eigenvalue problems are used (Ref 8).

The generalized eigenvalue problem to calculate transmission zeros depends on the number of inputs, m, and the number of outputs, ℓ, of the system.

For $\ell \geq m$ the problem is

$$\begin{bmatrix} \underline{A} & \underline{B} & 0 \\ \underline{C} & \underline{D} & 0 \\ 0 & 0 & \underline{K} \end{bmatrix} \underline{z} = \lambda \begin{bmatrix} I_n & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \underline{z} \tag{B.2}$$

For $m \geq \ell$ the problem is

$$\begin{bmatrix} \underline{A} & \underline{B} & 0 \\ \underline{C} & \underline{D} & 0 \\ 0 & I_m & \underline{K} \end{bmatrix} \underline{z} = \lambda \begin{bmatrix} I_n & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \underline{z} \tag{B.3}$$

The generalized eigenvalue problem to calculate decoupling zeros is

$$\begin{bmatrix} \underline{A} & \underline{B} & 0 \\ \underline{C} & \underline{D} & \underline{I}_{\ell} \\ 0 & I_m & \underline{K} \end{bmatrix} \underline{z} = \lambda \begin{bmatrix} I_n & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \underline{z} \tag{B.4}$$

In the above generalized eigenvalue problems, $\underline{k}$ is a random m - by - ℓ real matrix.

The generalized eigenvalue problem can be described as

$$\left[\underline{AA}\right] \underline{z} = \lambda \left[\underline{BB}\right] \underline{z} \tag{B.5}$$

The QZ algorithm has three steps. The first reduces the matrix $\underline{AA}$ to upper Hessenberg form and the matrix $\underline{BB}$ to upper triangular form. Next the Hessenberg matrix is further reduced to quasi-triangular form, while maintaining the triangular form of the other matrix. The final step reduces the quasi-triangular matrix further, such that the remaining 2 - by - 2 blocks correspond to pairs of complex eigenvalues. The QZ algorithm returns quantities whose ratio gives the generalized eigenvalues.

The computer program ZERO makes some decisions before the system zeros are finally classified. Refer to the flow chart shown in Figure B-1. For each path of the zero program, the appropriate generalized eigenvalue problem, G.E.P., is solved twice (runs 1 and 2) with only the $\underline{K}$ matrix changing. The outputs of runs 1 and 2 are compared. If the value of an eigenvalue is the same for both runs, then it is classified as a zero.

The zeros calculated by the QZ algorithm, set z_{TA}, may contain some decoupling zeros. Hence, set z_{TA} is compared with the decoupling zeros calculated by the QZ algorithm, set z_D. Those values in set z_{TA} that also exist in set z_D are eliminated from set z_{TA}. The remaining zeros in set z_{TA} form the set of transmission zeros, set z_T.

Experience shows that occasionally some values are incorrectly identified as system zeros. Hence, ZERO uses three different tests in order to filter out incorrect values from the sets z_T and z_D, and to further classify the real decoupling zeros.

Test A (Subroutine Syszero) uses singular value decomposition to check the rank of the system matrix

$$\underline{P}(\lambda) = \begin{bmatrix} \lambda I_n - \underline{A}, & -\underline{B} \\ \underline{C} & , \underline{D} \end{bmatrix} \qquad \text{(B.6)}$$

An invariant zero causes $\underline{P}(\lambda)$ to lose rank. All transmission zeros are invariant zeros, along with some decoupling zeros. All of the real transmission zeros and real decoupling zeros are sent through Test A. Any transmission zero that does not cause $\underline{P}(\lambda)$ to lose rank is rejected as a zero. Any decoupling zero that does cause $\underline{P}(\lambda)$ to lose rank is classified as an invariant decoupling zero.

Test B (Subroutine RDZ) uses singular value decomposition to check the rank of the matrices $\begin{bmatrix} \lambda I_n - \underline{A}, \underline{B} \end{bmatrix}$ and $\begin{bmatrix} \lambda I_n - \underline{A} \\ C \end{bmatrix}$. All of the real decoupling zeros are sent through Test B, in order to classify them as input, output, or input/output decoupling zeros. Input decoupling zeros cause the matrix $\begin{bmatrix} \lambda I_n - \underline{A}, \underline{B} \end{bmatrix}$ to be rank deficient. Output decoupling zeros

cause the matrix $\begin{bmatrix} \lambda I_n - \underline{A} \\ \underline{C} \end{bmatrix}$ to be rank deficient. Input/output decoupling zeros cause both matrices to be rank deficient. A real decoupling zero that causes neither matrix to lose rank is rejected as a zero.

Test C (Subroutine Complex) is for complex system zeros. All complex zeros must occur in conjugate pairs. Any complex zero without a conjugate is rejected.

ZERO is an interactive program. The required inputs are listed in Table B-1. Input data may be read from Tape3 = ZDATA. The order of the inputs for Tape3 = ZDATA is also listed in Table B-1. The ZDATA input format is (3I2, $(n^2+nm+2n\ell)$F20.7). Two ASD libraries, EISPACK and IMSL, must be connected before execution of the program ZERO.

The current version of ZERO is not 100% accurate. Sometimes an incorrect value will pass through the filter tests. These values can be easily identified because they are very large values. In the third example the number -19,722,484.728 is incorrectly identified as a transmission zero.

TABLE B-1

Required Input Data for Program Zero

Name	Symbol	Dimension	Input Order Interactive	Input Order ZDATA
Number of States	n		1	1
$\underline{A}$ Matrix	AMAT	n-by-n	2	4
Number of Inputs	m		3	2
$\underline{B}$ Matrix	BMAT	n-by-m	4	5
Number of Outputs			5	3
$\underline{C}$ Matrix	CMAT	ℓ-by-n	6	6
$\underline{D}$ Matrix	DMAT	ℓ-by-m	7	
$\underline{F}$ Matrix	FMAT	ℓ-by-n		7

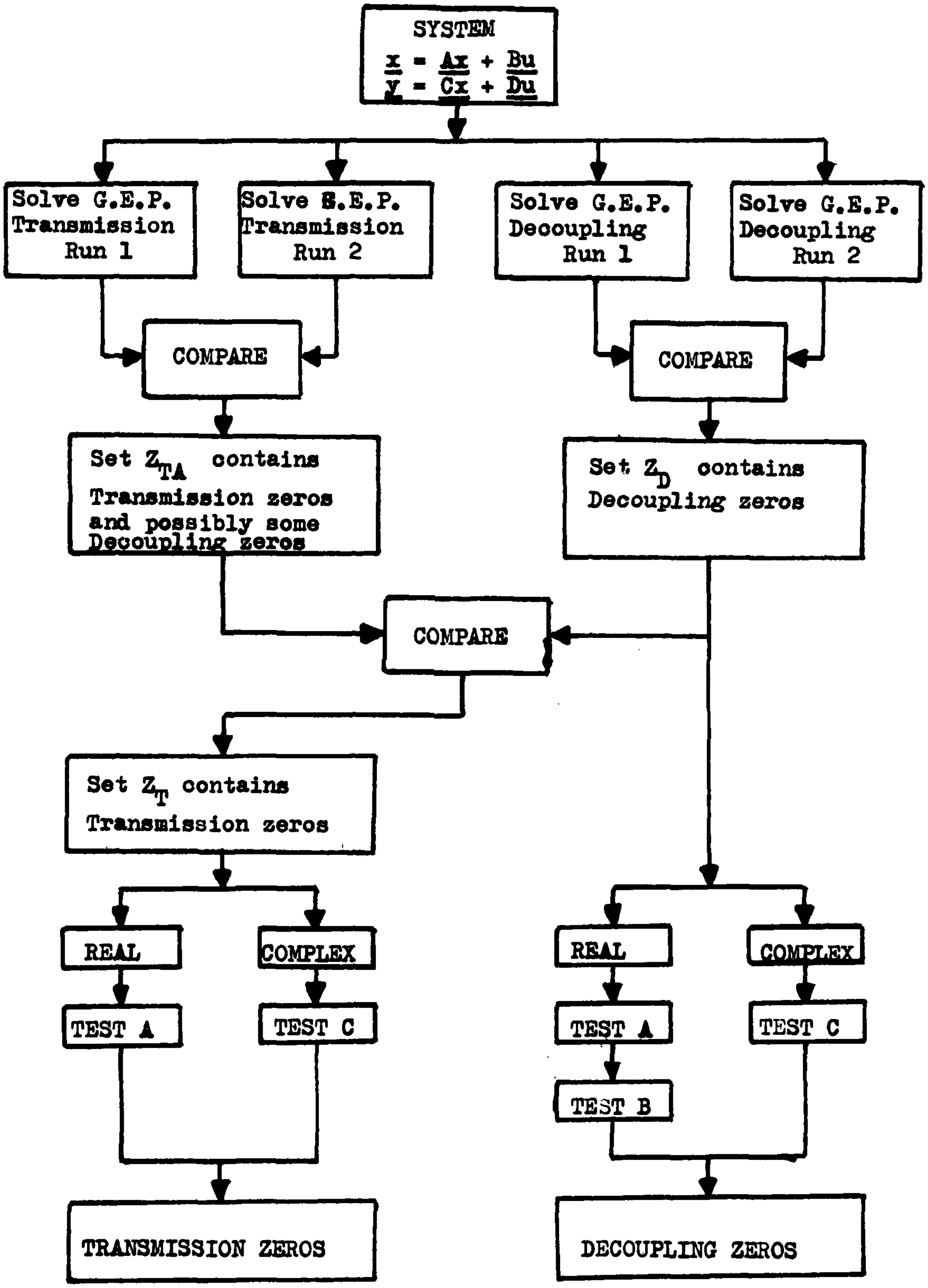


Figure B-1 Program ZERO Flow Chart

Appendix B-2

Listing of Program ZERO and Examples

Three examples are included in this appendix. The first two examples are from published papers (Ref 4:60) and (Ref 3:792). The third example is the system of design 2 used in this thesis.

```
THIS TEST PROGRAM CALCULATES ZEROS OF A LINEAR
MATRIX SYSTEM.
  MAXIMUM # OF STATES = 8.
            OF INPUTS = 4.
            OF OUTPUTS= 4.
TO INCREASE MAXIMUM, SEE INSTRUCTIONS AT THE
 BEGINNING OF THE PROGRAM LISTING.
FILE ZEROLIS,ID=LEWIS,SN=AFIT

ARE THE MATRICES STORED ON TAPE 3 = ZDATA ?
    ( 1 = YES.    0 = NO ) : 0

ENTER THE DIMENSION OF THE SYSTEM : 6

 ENTER THE A MATRIX (6 BY 6) :

ROW 1 : 1,0,0,0,0,0
ROW 2 : 0,1,0,0,0,0
ROW 3 : 0,0,3,0,0,0
ROW 4 : 0,0,0,-4,0,0
ROW 5 : 0,0,0,0,-1,0
ROW 6 : 0,0,0,0,0,3

ENTER THE NUMBER OF INPUTS : 2

 ENTER THE B MATRIX (6 BY 2)

ROW 1 : 0,-1
ROW 2 : -1,0
ROW 3 : 1,-1
ROW 4 : 0,0
ROW 5 : 0,1
ROW 6 : -1,-1

ENTER THE NUMBER OF OUTPUTS : 3

ENTER THE C MATRIX (3 BY 6)

ROW 1 : 1,0,0,1,0,0
ROW 2 : 0,1,0,1,0,1
ROW 3 : 0,0,1,0,0,1

IS THERE A D MATRIX ?
  (YES=1  NO=0) : 0

THE D MATRIX IS A (3 BY 2) ZERO MATRIX
```

HE A MATRIX IS :

1.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	1.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	3.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	-4.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	-1.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	3.0000

THE B MATRIX IS :

0.0000	-1.0000
-1.0000	0.0000
1.0000	-1.0000
0.0000	0.0000
0.0000	1.0000
-1.0000	-1.0000

THE C MATRIX IS :

1.0000	0.0000	0.0000	1.0000	0.0000	0.0000
0.0000	1.0000	0.0000	1.0000	0.0000	1.0000
0.0000	0.0000	1.0000	0.0000	0.0000	1.0000

```
*** WARNING ERROR            (IER =  33) FROM IMSL ROUTINE LSVDF
*** WARNING ERROR            (IER =  33) FROM IMSL ROUTINE LSVDF
    WARNING 33 MEANS A MATRIX DOES NOT
    HAVE FULL RANK.  THIS MESSAGE IS EXPECTED
```

THE TRANSMISSION ZEROS

REAL PART	IMMAGINARY PART
2.0000000000	0.

THE DECOUPLING ZEROS

-1.0000	OUTPUT INVARIANT
-4.0000	INPUT

DO YOU WISH TO CHANGE MATRICES
AND CONTINUE?

```
STOP = 0
CHANGE A ONLY    = 1
CHANGE B ONLY    = 2
CHANGE C ONLY    = 3
CHANGE A,B,C,D   = 4
CHANGE B,C ONLY  = 5
CHANGE C,D ONLY  = 6
: 4
ENTER THE DIMENSION OF THE SYSTEM : 6

 ENTER THE A MATRIX (6 BY 6) :

ROW 1 : 0,1,0,0,0,0
ROW 2 : 0,0,1,0,0,0
ROW 3 : 0,0,0,0,0,0
ROW 4 : 0,0,0,0,1,0
ROW 5 : 0,0,0,0,0,1
ROW 6 : 0,0,0,0,0,0

ENTER THE NUMBER OF INPUTS : 2

 ENTER THE B MATRIX (6 BY 2)

ROW 1 : 0,0
ROW 2 : 0,0
ROW 3 : 1,0
ROW 4 : 0,0
ROW 5 : 0,0
ROW 6 : 0,1

ENTER THE NUMBER OF OUTPUTS : 2

ENTER THE C MATRIX (2 BY 6)

ROW 1 : 1,1,0,0,0,0
ROW 2 : 0,0,0,-1,-1,0

IS THERE A D MATRIX ?
  (YES=1  NO=0) : 1

ENTER THE D MATRIX (2 BY 2)

ROW 1 : 1,0
ROW 2 : 1,0
```

THE A MATRIX IS :

0.0000	1.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	1.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

THE B MATRIX IS :

0.0000	0.0000
0.0000	0.0000
1.0000	0.0000
0.0000	0.0000
0.0000	0.0000
0.0000	1.0000

THE C MATRIX IS :

1.0000	1.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	-1.0000	-1.0000	0.0000

THE D MATRIX IS :

1.0000	0.0000
1.0000	0.0000

```
*** WARNING ERROR            (IER =  33) FROM IMSL ROUTINE LSVDF
*** WARNING ERROR            (IER =  33) FROM IMSL ROUTINE LSVDF
    WARNING 33 MEANS A MATRIX DOES NOT
    HAVE FULL RANK.  THIS MESSAGE IS EXPECTED
```

THE TRANSMISSION ZEROS

REAL PART	IMMAGINARY PART
.34116390191	1.1615414000
.34116390191	-1.1615414000
-1.000000(00	0.
-.68232780[illegible]	0.

THERE ARE NO DECOUPLING ZEROS

```
THIS TEST PROGRAM CALCULATES ZEROS OF A LINEAR
MATRIX SYSTEM.
  MAXIMUM # OF STATES = 8.
            OF INPUTS = 4.
            OF OUTPUTS= 4.
TO INCREASE MAXIMUM, SEE INSTRUCTIONS AT THE
 BEGINNING OF THE PROGRAM LISTING.
FILE ZEROLIS,ID=LEWIS,SN=AFIT

ARE THE MATRICES STORED ON TAPE 3 = ZDATA ?
    ( 1 = YES.   0 = NO ) : 1

THE A MATRIX IS :


 0.0000     0.0000     1.0000      .0318
  .0489     -.3744      .0318     -.9995
 0.0000   -39.9283    -2.1138      .0797
 0.0000     6.2379     -.0063     -.7074


THE B MATRIX IS :


 0.0000     0.0000     0.0000
  .0531      .0046      .0250
 9.7929   -55.7388     2.6033
-5.0544    -1.8927     4.3880


THE C MATRIX IS :


 0.0000     1.0000     0.0000     0.0000
 1.0000     0.0000     0.0000     0.0000
 0.0000     0.0000     0.0000     1.0000

*** WARNING ERROR           (IER =  33) FROM IMSL ROUTINE LSVDF
*** WARNING ERROR           (IER =  33) FROM IMSL ROUTINE LSVDF
    WARNING 33 MEANS A MATRIX DOES NOT
    HAVE FULL RANK.  THIS MESSAGE IS EXPECTED


THE TRANSMISSION ZEROS

   REAL PART        IMMAGINARY PART


-19722484.728       0.

THERE ARE NO DECOUPLING ZEROS
```

```
O YOU WISH TO CHANGE MATRICES
   AND CONTINUE?

   STOP = 0
   CHANGE A ONLY    = 1
   CHANGE B ONLY    = 2
   CHANGE C ONLY    = 3
   CHANGE A,B,C,D   = 4
   CHANGE B,C ONLY  = 5
   CHANGE C,D ONLY  = 6
 : 3
   CHANGE C-MATRIX TO F-MATRIX ?
   YES=1,   NO=0   : 1


   THE C MATRIX IS :


  0.0000      1.0000      0.0000      0.0000
  1.0000      0.0000       .2500       .0080
  0.0000      0.0000      0.0000      1.0000

*** WARNING ERROR            (IER =   33) FROM IMSL ROUTINE LSVDF
   WARNING 33 MEANS A MATRIX DOES NOT
   HAVE FULL RANK.  THIS MESSAGE IS EXPECTED


   THE TRANSMISSION ZEROS

      REAL PART          IMMAGINARY PART


 -4.0000000000        0.

   THERE ARE NO DECOUPLING ZEROS



   DO YOU WISH TO CHANGE MATRICES
   AND CONTINUE?

   STOP = 0
   CHANGE A ONLY    = 1
   CHANGE B ONLY    = 2
   CHANGE C ONLY    = 3
   CHANGE A,B,C,D   = 4
   CHANGE B,C ONLY  = 5
   CHANGE C,D ONLY  = 6
 : 0
```

```
      PROGRAM ZERO(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
     1   ZDATA,TAPE3=ZDATA)
      REAL TRAR(16),TRAI(16),DECR(16),DECI(16),FMAT(8,8)
      REAL AMAT(8,8),BMAT(8,8),CMAT(8,8),DMAT(8,8)
      INTEGER IFLT(16),IFL(16),IO(16)
C
      LOGICAL MATV
C
C        THIS PROGRAM CAN BE SIMPLY ALTERED TO ACCOMMIDATE
C        MORE STATES AND INPUTS/OUTPUTS.
C        THIS IS DONE BY CHANGING DIMENSION STATEMENTS
C        THE IMPORTANT DIMENSION NUMBERS ARE:
C          N - NUMBER OF STATES
C          M - NUMBER OF INPUTS
C          L - NUMBER OF OUTPUTS
C          NM - N + MAX(M,L)
C          NN - N + M + L
C
C        CHANGE ALL ARRAYS WITH (8,8) TO (N,N)
C        CHANGE ALL ARRAYS WITH (12,12) TO (NM,NM)
C        CHANGE ,12, IN IMSL FUNCTION LSVDF IN SUBROTINE RDZ
C       AND SUBROUTINE SYSZERO TO ,NM.
C        CHANGE ALL VECTORS WITH (16) AND ARRAYS
C         WITH (16,16) TO (NN) OR (NN,NN)
C        ALSO CHANGE THE ,16, IN THE QZ SUBROTINES
C       TO ,NN.
C
C         A SAMPLE COMMAND IN EDITOR IS : /16/=/NN/,A,U,V (VETO)
C      SAVE NEW FILE THEN RECOMPLILE
C
C
      PRINT*," "
      PRINT*,"    THIS TEST PROGRAM CALCULATES ZEROS OF A LINEAR"
      PRINT*,"    MATRIX SYSTEM."
      PRINT*,"      MAXIMUM # OF STATES = 8."
      PRINT*,"                OF INPUTS = 4."
      PRINT*,"                OF OUTPUTS= 4."
      PRINT*,"    TO INCREASE MAXIMUM, SEE INSTRUCTIONS AT THE"
      PRINT*,"     BEGINNING OF THE PROGRAM LISTING."
      PRINT*,"    FILE ZEROLIS,ID=LEWIS,SN=AFIT"
      PRINT*," "
      PRINT*,"    ARE THE MATRICES STORED ON TAPE 3 = ZDATA ?"
      PRINT*,"        ( 1 = YES.   0 = NO ) : "
      READ*,N1
      IF(N1.EQ.1) GO TO 1100
C
      PRINT*," "
      MATCH=0
 5400 PRINT*,"    ENTER THE DIMENSION OF THE SYSTEM :"
      READ*,N
C
C      INPUT THE BLOCK A MATRIX
C
      PRINT*," "
```

```
      PRINT*,"     ENTER THE A MATRIX (",N," BY ",N,") :"
      CALL READM(N,N,AMAT)
      IF(MATCH.EQ.1)GO TO 5430
C
      PRINT*," "
 5410 PRINT*,"    ENTER THE NUMBER OF INPUTS :"
      READ*,M
C
C      INPUT THE BLOCK B MATRIX
C
      PRINT*," "
      PRINT*,"     ENTER THE B MATRIX (",N," BY ",M,")"
      CALL READM(N,M,BMAT)
      IF(MATCH.EQ.2)GO TO 5430
C
      PRINT*," "
 5420 IF(N1.EQ.1) GO TO 5421
 5424 PRINT*,"    ENTER THE NUMBER OF OUTPUTS :"
      READ*,L
      GO TO 5422
 5421 PRINT*,"    CHANGE C-MATRIX TO F-MATRIX ?"
      PRINT*,"    YES=1,  NO=0  :"
      READ*,N2
      IF(N2.EQ.1) GO TO 5423
      GO TO 5424
 5422 CONTINUE
C
C      INPUT THE BLOCK C MATRIX
C
      PRINT*," "
      PRINT*,"    ENTER THE C MATRIX (",L," BY ",N,")"
      CALL READM(L,N,CMAT)
C
      IF(MATCH.EQ.4) GO TO 5435
      IF(MATCH.EQ.6) GO TO 5435
      IF(MATCH.NE.0) GO TO 1200
C
 5435 PRINT*," "
      PRINT*,"    IS THERE A D MATRIX ?"
      PRINT*,"      (YES=1  NO=0) :"
      READ*,I9
      IF(I9.EQ.1) GO TO 5440
C
      DO 5450 I=1,L
      DO 5450 J=1,M
 5450 DMAT(I,J)=0.0
      PRINT*," "
      PRINT*,"    THE D MATRIX IS A (",L," BY ",M,") ZERO MATRIX"
      PRINT*," "
      GO TO 5430
C
 5440 PRINT*," "
      PRINT*,"    ENTER THE D MATRIX (",L," BY ",M,")"
      CALL READM(L,M,DMAT)
```

```
C
 5430 GO TO 1200
C
C
C         READ THE INPUT DATA FROM TAPE 3 = ZDATA
C
C
 1100 CONTINUE
C
      READ(3,45) N,M,L
   45 FORMAT(3I2)
C
      DO 50 I=1,N
      DO 50 J=1,N
   50 READ(3,55) AMAT(I,J)
C
      DO 60 I=1,N
      DO 60 J=1,M
   60 READ(3,55) BMAT(I,J)
C
      DO 70 I=1,L
      DO 70 J=1,N
   70 READ(3,55) CMAT(I,J)
C
      DO 80 I=1,L
      DO 80 J=1,N
   80 READ(3,55) FMAT(I,J)
C
       I9=0
      DO 90 I=1,L
      DO 90 J=1,M
   90 DMAT(I,J)=0.0
C
C
   55 FORMAT(F20.7)
      GO TO 1200
C
 5423 CONTINUE
      DO 85 I=1,L
      DO 85 J=1,N
   85 CMAT(I,J)=FMAT(I,J)
C
 1200 CONTINUE
C
C
       IF(MATCH.EQ.2) GO TO 91
       IF(MATCH.EQ.3) GO TO 92
       IF(MATCH.EQ.5) GO TO 91
       IF(MATCH.EQ.6) GO TO 92
C
       PRINT*," "
       PRINT*,"    THE A MATRIX IS :"
       PRINT*," "
      CALL PRINTM(N,N,AMAT)
```

```
C
      IF(MATCH.EQ.1) GO TO 95
C
   91 PRINT*," "
      PRINT*," "
      PRINT*,"    THE B MATRIX IS :"
      PRINT*," "
      CALL PRINTM(N,M,BMAT)
C
      IF(MATCH.EQ.2) GO TO 95
C
   92 PRINT*," "
      PRINT*," "
      PRINT*,"    THE C MATRIX IS :"
      PRINT*," "
      CALL PRINTM(L,N,CMAT)
      PRINT*," "
      IF(MATCH.EQ.3) GO TO 95
      IF(MATCH.EQ.5) GO TO 95
C
      IF(I9.EQ.1) GO TO 96
      GO TO 95
   96 PRINT*," "
      PRINT*,"    THE D MATRIX IS :"
      PRINT*," "
      CALL PRINTM(L,M,DMAT)
      PRINT*," "
   95 CONTINUE
C
C
C           CALCULATE ZEROS
C
C
      NN=N+M+L
      MATV=.FALSE.
C
C
C       SUBROUTINE TZ CALCULATES THE TRANSMISSION ZEROS
C
C
      CALL TZ(AMAT,BMAT,CMAT,DMAT,NN,N,M,L,MATV,
     +TRAR,TRAI,IFLT)
C
C       DOUBLE CHECK REAL TRANSMISSION ZEROS BY TESTING
C       THE RANK OF THE SYSTEM MATRIX.  A TRANSMISSION
C       ZERO CAUES THE SYSTEM MATRIX TO BE RANK DEFICIENT.
C
      EX3=1.0E-3
C
      DO 440 I=1,NN
      IF(IFLT(I).EQ.0)GO TO 440
      IF(TRAI(I).LT.EX3.AND.TRAI(I).GT.-EX3) GO TO 445
      GO TO 440
  445 Z=TRAR(I)
```

```
      CALL SYSZERO(AMAT,BMAT,CMAT,DMAT,Z,N,M,L,IT,1)
      IF(IT.EQ.0) IFLT(I)=0
  440 CONTINUE
C
C       SUBROUTINE DZ CALCULATES THE DECOUPLING ZEROS
C
C
      CALL DZ(AMAT,BMAT,CMAT,DMAT,NN,N,M,L,MATV,
     +DECR,DECI,IFL)
C
C
C      COMPARE THE TRANSMISSION ZEROS FOUND IN TZ
C      WITH THE DECOUPLING ZEROS FOUND IN DZ.
C      IF THE SAME VALUE APPEARS IT IS ONLY A
C      DECOUPLING ZERO.
C
      EX=1.0E-5
C
      DO 200 I=1,NN
C
      IF(IFLT(I).EQ.0) GO TO 200
      ATRAR=ABS(TRAR(I))
      ATRAI=ABS(TRAI(I))
C
      DO 210 J=1,NN
C
      IF(IFL(J).EQ.0) GO TO 210
      ADECR=ABS(DECR(J))
      ADECI=ABS(DECI(J))
C
C      COMPARE REAL SYSTEM ZEROS
C
      IF(ATRAI.GT.EX3) GO TO 201
      IF(ADECI.GT.EX3) GO TO 210
      IF(ATRAR.LT.EX3) GO TO 202
C
      DT=(TRAR(I)-DECR(J))/TRAR(I)
      IF(ABS(DT).LT.EX) IFLT(I)=0
      GO TO 210
C
  202 IF(TRAR(I).LT.DECR(J)+EX.AND.TRAR(I).GT.DECR(J)-EX) IFLT(I)=0
      GO TO 210
C
C      COMPARE IMMANGINARY SYSTEM ZEROS
C
  201 IF(ADECI.LT.EX3) GO TO 210
      IF(ATRAR.LT.EX3) GO TO 203
C
      DTI=(TRAR(I)-DECR(J))/TRAR(I)
      IF(ABS(DTI).LT.EX) GO TO 204
      GO TO 210
C
  203 IF(TRAR(I).LT.DECR(J)+EX.AND.TRAR(I).GT.DECR(J)-EX) GO TO 204
      GO TO 210
```

```
C
 204 IF(ATRAI.LT.EX3) GO TO 205
     DTII=(TRAI(I)-DECI(J))/TRAI(I)
     IF(ABS(DTII).LT.EX) IFLT(I)=0
     GO TO 210
C
 205 IF(TRAI(I).LT.DECI(J)+EX.AND.TRAI(I).GT.DECI(J)-EX) IFLT(I)=0
C
 210 CONTINUE
 200 CONTINUE
C
C      CHECK CALCULATED COMPLEX ZEROS FOR COMPLEX
C      CONJIGATE PAIRS. COMPLEX ZEROS MUST OCCUR IN
C      CONJIGATE PAIRS.
C
     CALL COMPLEX(NN,TRAI,IFLT)
     CALL COMPLEX(NN,DECI,IFL)
C
     KTZ=0
     ITZ=1
     DO 300 I=1,NN
     IF(IFLT(I).EQ.1) GO TO 300
     KTZ=KTZ+1
     IF(KTZ.EQ.NN) GO TO 305
     GO TO 300
 305 ITZ=0
 300 CONTINUE
C
     KDZ=0
     IDZ=1
     DO 310 J=1,NN
     IF(IFL(J).EQ.1)GO TO 310
     KDZ=KDZ+1
     IF(KDZ.EQ.NN) GO TO 315
     GO TO 310
 315 IDZ=0
 310 CONTINUE
C
C
C     TEST DECOUPLING ZEROS TO DETERMIN WETHER THEY
C     ARE INVARIANT, AND IF THEY ARE AN INPUT OR OUTPUT
C     ZERO.
C
C
     DO 500 I=1,NN
     IO(I)=0
     IF(IFL(I).EQ.0) GO TO 500
     IF(DECI(I).LT.EX3.AND.DECI(I).GT.-EX3) GO TO 550
     GO TO 500
 550 Z=DECR(I)
     CALL RDZ(AMAT,BMAT,CMAT,Z,N,M,L,KIO)
     CALL SYSZERO(AMAT,BMAT,CMAT,DMAT,Z,N,M,L,ID,2)
     IF(ID.EQ.1) GO TO 510
     IF(KIO.EQ.1) IO(I)=1
```

```
      IF(KIO.EQ.2) IO(I)=2
      IF(KIO.EQ.3) IO(I)=3
      GO TO 500
  510 IF(KIO.EQ.1) IO(I)=4
      IF(KIO.EQ.2) IO(I)=5
      IF(KIO.EQ.3) IO(I)=6
  500 CONTINUE
C
      IF(ITZ.EQ.0.AND.IDZ.EQ.0) GO TO 399
      PRINT*,"     WARNING 33 MEANS A MATRIX DOES NOT"
      PRINT*,"     HAVE FULL RANK.  THIS MESSAGE IS EXPECTED"
      PRINT*," "
      PRINT*," "
  399 CONTINUE
C
C
      IF(ITZ.EQ.0) GO TO 400
      PRINT*," "
      PRINT*,"     THE TRANSMISSION ZEROS"
      PRINT*," "
      PRINT 412
      PRINT*," "
C
      DO 410 I=1,NN
      IF(IFLT(I).EQ.0) GO TO 410
      PRINT*," "
      PRINT 411,(TRAR(I),TRAI(I))
  410 CONTINUE
C
  415 IF(IDZ.EQ.0) GO TO 405
      PRINT*," "
      PRINT*,"     THE DECOUPLING ZEROS"
C
      DO 420 I=1,NN
      IF(IFL(I).EQ.0) GO TO 420
      IF(IO(I).EQ.0) GO TO 421
      IF(IO(I).EQ.1) GO TO 422
      IF(IO(I).EQ.2) GO TO 423
      IF(IO(I).EQ.4) GO TO 424
      IF(IO(I).EQ.5) GO TO 426
      IF(IO(I).EQ.6) GO TO 427
C
      PRINT*," "
      PRINT 413,DECR(I)
      GO TO 420
  427 PRINT*," "
      PRINT 417,DECR(I)
      GO TO 420
  426 PRINT*," "
      PRINT 418,DECR(I)
      GO TO 420
  424 PRINT*," "
      PRINT 419,DECR(I)
      GO TO 420
```

```
 423 PRINT*," "
     PRINT 414,DECR(I)
     GO TO 420
 422 PRINT*," "
     PRINT 416,DECR(I)
     GO TO 420
 421 PRINT*," "
     PRINT 412
     PRINT 411,(DECR(I),DECI(I))
 420 CONTINUE
C
 411 FORMAT(2G20.11)
 412 FORMAT(8X,"REAL PART",8X,"IMMAGINARY PART")
 413 FORMAT(5X,F10.4,5X," INPUT/OUTPUT")
 414 FORMAT(5X,F10.4,5X," OUTPUT")
 416 FORMAT(5X,F10.4,5X," INPUT")
 417 FORMAT(5X,F10.4,5X," INPUT/OUTPUT  INVARIANT")
 418 FORMAT(5X,F10.4,5X," OUTPUT  INVARIANT")
 419 FORMAT(5X,F10.4,5X," INPUT  INVARIANT")
C
C
C
     GO TO 430
 400 PRINT*," "
     PRINT*,"     THERE ARE NO TRANSMISSION ZEROS"
     PRINT*," "
     PRINT*," "
     PRINT*," "
     GO TO 415
 405 PRINT*," "
     PRINT*,"     THERE ARE NO DECOUPLING ZEROS"
     PRINT*," "
     PRINT*," "
     PRINT*," "
 430 CONTINUE
C
C
C       SELECT OPTION. CONTINUE OR RETURN
C
     PRINT*," "
     PRINT*,"     DO YOU WISH TO CHANGE MATRICES"
     PRINT*,"     AND CONTINUE?"
     PRINT*," "
     PRINT*,"     STOP = 0"
     PRINT*,"     CHANGE A ONLY   = 1"
     PRINT*,"     CHANGE B ONLY   = 2"
     PRINT*,"     CHANGE C ONLY   = 3"
     PRINT*,"     CHANGE A,B,C,D  = 4"
     PRINT*,"     CHANGE B,C ONLY = 5"
     PRINT*,"     CHANGE C,D ONLY = 6"
     PRINT*,"  :"
     READ*,MATCH
     IF(MATCH.EQ.1) GO TO 5400
     IF(MATCH.EQ.2) GO TO 5410
```

```
      IF(MATCH.EQ.3) GO TO 5420
      IF(MATCH.EQ.4) GO TO 5400
      IF(MATCH.EQ.5) GO TO 5410
      IF(MATCH.EQ.6) GO TO 5420
      STOP
      END
C
C
      SUBROUTINE READM(L,N,A)
      REAL A(8,8)
      PRINT*," "
C
      DO 100 I=1,L
      PRINT*,"    ROW ",I," :"
      READ*,(A(I,J),J=1,N)
  100 CONTINUE
C
      RETURN
      END
C
      SUBROUTINE PRINTM(L,N,A)
      REAL A(8,8)
      PRINT*," "
C
      DO 100 I=1,L
      PRINT 200,(A(I,J),J=1,N)
  100 CONTINUE
C
  200 FORMAT(1X,8(F12.4,3X))
      RETURN
      END
C
C
      SUBROUTINE MSUB(A,B,C,L,N)
      REAL A(8,8),B(8,8),C(8,8)
C
      DO 100 I=1,L
      DO 100 J=1,N
  100 C(I,J)=A(I,J)-B(I,J)
C
      RETURN
      END
C
C
      SUBROUTINE ABCD(AA,BB,A,B,C,D,NN,N,M,L)
      REAL AA(16,16),BB(16,16),A(8,8),B(8,8),
     +C(8,8),D(8,8)
C
C      THIS SUBROUTINE SETS UP THE MATRICES AA AND BB
C      WHERE AA=( A,B ), AND BB= ( I,0 )
C               ( C,D ),         ( 0,0 )
C
      DO 200 I=1,NN
      DO 200 J=1,NN
```

```
      AA(I,J)=0.
  200 BB(I,J)=0.
C
      DO 210 I=1,N
  210 BB(I,I)=1.
C
      DO 220 I=1,N
      DO 220 J=1,N
  220 AA(I,J)=A(I,J)
C
      DO 230 I=1,N
      DO 230 J=1,M
  230 AA(I,N+J)=B(I,J)
C
      DO 240 I=1,L
      DO 240 J=1,N
  240 AA(N+I,J)=C(I,J)
C
      DO 250 I=1,L
      DO 250 J=1,M
  250 AA(N+I,N+J)=D(I,J)
C
      RETURN
      END
C
C
      SUBROUTINE QZALGM(N,A,B,ALFR,ALFI,BETA,
     +MATZ,Z,IFAIL)
      LOGICAL MATZ
C
C      THIS SUBROUTINE CALLS THE QZ SUBROUTINES FROM THE
C       SLATEC (OR EISPACK) LIBRARY
C
      REAL A(16,16),B(16,16),ALFR(16),ALFI(16),BETA(16),Z(16,16)
      ISAVE=IFAIL
      IFAIL=1
      ESP=-1.0E-6
      CALL QZHES(16,N,A,B,MATZ,Z)
      CALL QZIT(16,N,A,B,ESP,MATZ,Z,IFAIL)
      CALL QZVAL(16,N,A,B,ALFR,ALFI,BETA,MATZ,Z)
      IF(IFAIL.NE.0)RETURN
      IF(MATZ)CALL QZVEC(16,N,A,B,ALFR,ALFI,BETA,Z)
      RETURN
      END
C
      SUBROUTINE CAZERO(AA,BB,NN,VV,MATV,ZERR,ZERI,
     +TR,TI,LABR,LABI,IFL,JF,IX)
C
C
      REAL AA(16,16),BB(16,16),ALFR(16),ALFI(16),BETA(16),
     +VV(16,16),LABR(16),LABI(16),ZERR(16),ZERI(16),
     +TR(16),TI(16)
      INTEGER IFL(16)
      LOGICAL MATV
```

```
C
C      THIS SUBROUTINE TAKES DATA COMPUTED BY THE QZ
C      ALGORITHM AND CALCULATES THE SYSTEM ZEROS. IT
C      ALSO DECIDES IF THE NUMBER IS OR ISN'T A SYSTEM
C      ZERO. THIS IS DONE BY USING THE QZ ALGORITHM
C      TWICE AND COMPARING THE OUTPUTS OF EACH RUN. IF
C      THE VALUE IS SAME FOR BOTH RUNS, IT IS A SYSTEM
C      ZERO.
C
C
C        THE VALUE OF IX SIGNALS WHICH SYSTEM ZERO
C        IS BEING CALCULATED.
C
C        TRANSMISSION ZEROS  IX=1
C        DECOUPLING ZEROS    IX=2
C
      IF(IX.EQ.1) EX1=1.0E-8
      IF(IX.EQ.2) EX1=1.0E-4
C
      EX2=1.0E-8
C
      IFAIL=1
      CALL QZALGM(NN,AA,BB,ALFR,ALFI,BETA,
     +MATV,VV,IFAIL)
      IF(IFAIL.EQ.0) GOTO 150
      PRINT*,"    RUN FAILED"
  150 CONTINUE
C
      DO 120 I=1,NN
      LABR(I)=0.
      IF(JF.EQ.1) TR(I)=0.
      LABI(I)=0.
      IF(JF.EQ.1) TI(I)=0.
      BT=BETA(I)
      IFL(I)=0
      IF(ABS(BT).GE.EX2) GOTO 40
      GO TO 120
   40 IFL(I)=1
      LABR(I)=ALFR(I)/BT
      LABI(I)=ALFI(I)/BT
      IF(JF.EQ.2.) GOTO 120
      TR(I)=LABR(I)
      TI(I)=LABI(I)
  120 CONTINUE
      IF(JF.EQ.2) GOTO 50
      GOTO 1000
   50 CONTINUE
C
C       COMPARE CALCULATED ZEROS FROM RUNS 1 AND 2
C
      DO 130 I=1,NN
      IF(IFL(I).EQ.0) GOTO 130
      ZERR(I)=LABR(I)
      ZERI(I)=LABI(I)
```

```
C
         AZR=ABS(ZERR(I))
         AZI=ABS(ZERI(I))
C
         K=0
C
         DO 42 J=1,NN
C
         ATR=ABS(TR(J))
         ATI=ABS(TI(J))
C
C
C         COMPARE REAL ZEROS
C
         IF(AZI.GT.EX2) GO TO 400
         IF(ATI.GT.EX2) GO TO 43
         IF(AZR.GT.EX2) GO TO 310
         IF(ZERR(I).LT.TR(J)+EX1.AND.ZERR(I).GT.TR(J)-EX1) GO TO 42
         GO TO 43
C
     310 DT=(ZERR(I)-TR(J))/ZERR(I)
         IF(ABS(DT).LT.EX1) GO TO 42
         GO TO 43
C
C         COMPARE IMMANGINARY ZEROS
C
     400 IF(ATI.LT.EX2) GO TO 43
         IF(AZR.GT.EX2) GO TO 410
         IF(ZERR(I).LT.TR(J)+EX1.AND.ZERR(I).GT.TR(J)-EX1) GO TO 420
         GO TO 43
C
     410 DT=(ZERR(I)-TR(J))/ZERR(I)
         IF(ABS(DT).LT.EX1) GO TO 420
         GO TO 43
C
     420 IF(AZI.GT.EX2) GO TO 430
         IF(ZERI(I).LT.TI(J)+EX1.AND.ZERI(I).GT.TI(J)-EX1) GO TO 42
         GO TO 43
C
     430 DTI=(ZERI(I)-TI(J))/ZERI(I)
         IF(ABS(DTI).LT.EX1) GO TO 42
C
      43 K=K+1
      42 CONTINUE
         IF(K.EQ.NN) IFL(I)=0
     130 CONTINUE
C
    1000 CONTINUE
         RETURN
         END
C
C
         SUBROUTINE TZ(A,B,C,D,NN,N,M,L,MATV,TRAR,TRAI,IFLT)
C
```

```
C
      REAL AA(16,16),BB(16,16),A(8,8),B(8,8),C(8,8),D(8,8),
     +TRAR(16),TRAI(16),TR(16),TI(16),LABR(16),LABI(16),VV(16,16)
      INTEGER IFLT(16)
C
      LOGICAL MATV
C
C      THIS SUBROUTINE AND THE SUBROUTINES CALLED BY IT
C      COMPUTE THE TRANSMISSION ZEROS OF A SYSTEM
C
      DO 100 II=1,2
      CALL ABCD(AA,BB,A,B,C,D,NN,N,M,L)
      IF(L.LT.M) GO TO 200
C
      DO 110 I=1,L
  110 AA(N+I,N+M+I)=1.
C
      GO TO 300
C
  200 CONTINUE
C
      DO 210 I=1,M
  210 AA(N+L+I,N+I)=1.
C
  300 CONTINUE
C
      DO 220 I=1,M
      DO 220 J=1,L
  220 AA(N+L+I,N+M+J)=RANF(DUM)*10.
C
C
      JF=II
      CALL CAZERO(AA,BB,NN,VV,MATV,TRAR,TRAI,
     +TR,TI,LABR,LABI,IFLT,JF,1)
  100 CONTINUE
C
      RETURN
      END
C
      SUBROUTINE DZ(A,B,C,D,NN,N,M,L,MATV,DECR,
     +DECI,IFL)
C
C
      INTEGER IFL(16)
C
C        THIS SUBROUTINE AND THE SUBROUTINES CALLED BY
C        IT COMPUTE THE DECOUPLING ZEROS OF A SYSTEM.
C
      REAL AA(16,16),BB(16,16),A(8,8),B(8,8),C(8,8)
     +,D(8,8),TR(16),TI(16),VV(16,16),
     2LABR(16),LABI(16),DECR(16),DECI(16)
      LOGICAL MATV
C
      DO 100 II=1,2
```

```
      CALL ABCD(AA,BB,A,B,C,D,NN,N,M,L)
C
      DO 110 I=1,L
  110 AA(N+I,N+M+I)=1.
C
      DO 120 I=1,M
  120 AA(N+L+I,N+I)=1.
C
      DO 130 I=1,M
      DO 130 J=1,L
  130 AA(N+L+I,N+M+J)=RANF(DUM)*10.
C
C
      JF=II
      CALL CAZERO(AA,BB,NN,VV,MATV,DECR,DECI,
     +TR,TI,LABR,LABI,IFL,JF,2)
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE RDZ(A,B,C,Z,N,M,L,KIO)
      REAL A(8,8),B(8,8),C(8,8),S(8),SO(8)
      REAL U(12,12),UO(12,12),SI(8,8),SIA(8,8)
      REAL DCIN(12,12),DCOT(12,12),WK(12),WKO(12)
C
C     THIS SUBROUTINE DETERMINES WEATHER A DECOUPLING ZERO
C      IS AN INPUT,OUTPUT,OR INPUT/OUTPUT DECOUPLING
C      ZERO. THIS IS DONE BY TAKING THE SINGULAR VALUE
C      DECOMPOSITION OF THE MATRICES (SI-A,B) AND (SI-A)
C                                                 (  C )
C      TO TEST FOR FULL RANK
C
      MO=N+M
      LO=N+L
C
      EX1=1.0E-1
      EX2=1.0E-5
C
      DO 190 I=1,N
      DO 190 J=1,N
  190 SI(I,J)=0.
C
      DO 200 I=1,N
  200 SI(I,I)=Z
C
      CALL MSUB(SI,A,SIA,N,N)
C
      DO 210 I=1,N
      DO 210 J=1,N
      DCIN(I,J)=SIA(I,J)
  210 DCOT(I,J)=SIA(I,J)
C
      DO 220 I=1,N
      DO 220 J=1,M
```

```
  220 DCIN(I,J+N)=B(I,J)
C
      DO 230 I=1,L
      DO 230 J=1,N
  230 DCOT(I+N,J)=C(I,J)
C
      CALL LSVDF(DCIN,12,N,MO,U,12,0,S,WK,IER)
C
      KIO=0
      KI=0
      DO 300 I=1,N
      IF(S(I).GT.EX1) GO TO 300
      IF(S(I).LT.-EX1) GO TO 300
      IF(S(I).LT.EX2.AND.S(I).GT.-EX2) GO TO 350
      GO TO 300
  350 KI=1
  300 CONTINUE
C
      CALL LSVDF(DCOT,12,LO,N,UO,12,0,SO,WKO,IERO)
C
      KOT=0
      DO 310 I=1,N
      IF(SO(I).GT.EX1) GO TO 310
      IF(SO(I).LT.-EX1) GO TO 310
      IF(SO(I).LT.EX2.AND.SO(I).GT.-EX2) GO TO 360
      GO TO 310
  360 KOT=1
  310 CONTINUE
C
      IF(KI.EQ.1) KIO=1
      IF(KOT.EQ.1) KIO=2
      IF(KI.EQ.1.AND.KOT.EQ.1) KIO=3
      RETURN
      END
C
C
      SUBROUTINE SYSZERO(A,B,C,D,Z,N,M,L,IZ,IX)
      REAL A(8,8),B(8,8),C(8,8),D(8,8),S(8),WK(12)
      REAL SYSM(12,12),SI(8,8),SIA(8,8),U(12,12)
C
C         THIS SUBROUTINE DETERMINS THE INVARIANT ZEROS
C         ALL TRANSMISSION ZEROS ARE INVARIANT
C         SOME DECOUPLING ZEROS ARE INVARIANT
C         THE SUBROUTINE TAKES THE SYSTEM MATRIX  ( SI-A , -B)
C                                                 (  C   ,  D)
C         AND TESTS FOR FULL RANK BY SINGULAR VALUE DECOMPOSITION
C         AN INVARIANT ZERO CAUSES THE SYSTEM MATIX TO LOOSE RANK
C
      IZ=0
C
      MO=N+M
      LO=N+L
C
      IF(IX.EQ.2) GO TO 20
```

```
C
      EX1=1.0E-1
      EX2=1.0E-9
      GO TO 25
C
   20 EX1=1.0E-1
      EX2=1.0E-5
   25 CONTINUE
C
      DO 100 I=1,LO
      DO 100 J=1,MO
  100 SYSM(I,J)=0.0
C
      DO 110 I=1,N
      DO 110 J=1,M
  110 SYSM(I,N+J)=-B(I,J)
C
      DO 120 I=1,L
      DO 120 J=1,N
  120 SYSM(N+I,J)=C(I,J)
C
      DO 130 I=1,L
      DO 130 J=1,M
  130 SYSM(N+I,N+J)=D(I,J)
C
C
      DO 200 I=1,N
      DO 200 J=1,N
  200 SI(I,J)=0.0
C
      DO 210 I=1,N
  210 SI(I,I)=Z
C
      CALL MSUB(SI,A,SIA,N,N)
C
      DO 220 I=1,N
      DO 220 J=1,N
  220 SYSM(I,J)=SIA(I,J)
C
      CALL LSVDF(SYSM,12,LO,MO,U,12,0,S,WK,IER)
C
      IF(LO.GT.MO) GO TO 320
C
      DO 300 I=1,LO
      IF(S(I).GT.EX1) GO TO 300
      IF(S(I).LT.-EX1) GO TO 300
      IF(S(I).LT.EX2.AND.S(I).GT.-EX2) IZ=1
  300 CONTINUE
C
      GO TO 330
C
  320 DO 310 I=1,MO
      IF(S(I).GT.EX1) GO TO 310
      IF(S(I).LT.-EX1) GO TO 310
```

```
      IF(S(I).LT.EX2.AND.S(I).GT.-EX2) IZ=1
  310 CONTINUE
C
  330 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE COMPLEX(NN,ZERI,IFL)
      REAL ZERI(16)
      INTEGER IFL(16)
C
C       THIS SUBROTINE TESTS FOR COMPLEX CONJIGATE PAIRS.
C       A COMPLEX SYSTEM ZERO WILL ALWAYS OCCUR IN CONJIGATE PAIRS
C       ANY COMPLEX ZERO CALCULATED WITHOUT A CONJIGATE PAIR WILL BE
C       REJECTED AS A SYSTEM ZERO
C
      EX1=1.0E-6
C
      DO 100 I=1,NN
      IF(IFL(I).EQ.0) GO TO 100
      IF(ZERI(I).LT.EX1.AND.ZERI(I).GT.-EX1) GO TO 100
C
      DO 110 J=1,NN
      IF(IFL(J).EQ.0) GO TO 110
      IF(ZERI(J).LT.EX1.AND.ZERI(J).GT.-EX1) GO TO 110
C
      CCP=ZERI(I)
      CCN=ZERI(J)
      CCC=CCP+CCN
      IF(CCC.LT.EX1.AND.CCC.GT.-EX1) GO TO 120
      GO TO 130
  120 IFL(I)=1
      IFL(J)=1
      GO TO 100
  130 IFL(I)=0
C
  110 CONTINUE
  100 CONTINUE
C
      RETURN
      END
*EOR
*EOF
```

Appendix C

Main Computer Program

This appendix contains a listing of the computer program used to design controllers and simulate them. The program is adopted from the program used by Lt Brett Ridgely (Ref 10). A simplified flow chart is shown in Figure C-1. The program listed is for designs with actuator dynamics. For designs without actuator actuator dynamics lines 2840 through 3450 are replaced by the lines 3300 through 4380 shown at the end of the main program listing.

The example is design 2A simulated at the design condition (5,000 ft, Mach 0.6) and also at the condition of design 3 (10,000 ft, Mach 0.9).

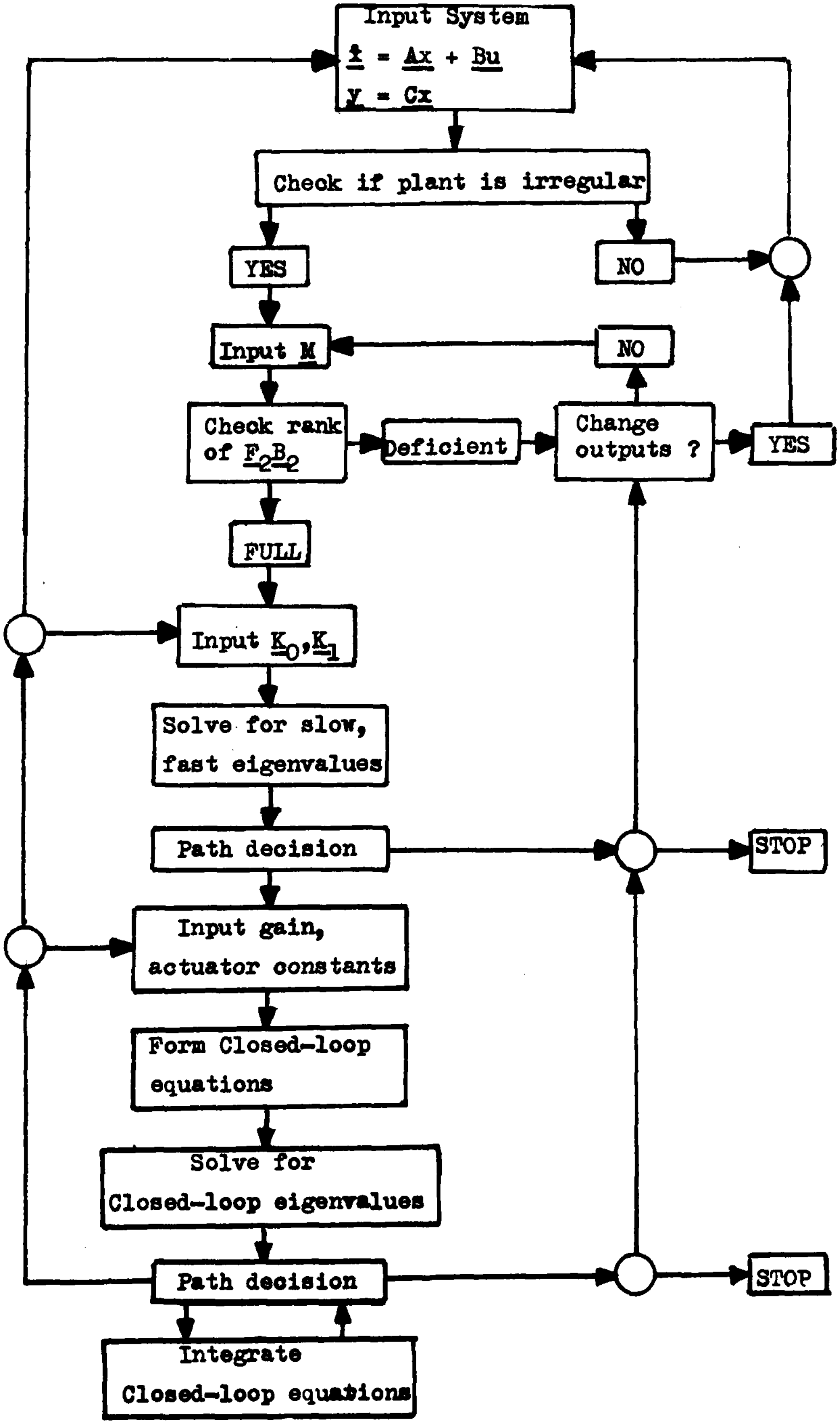


Figure C-1 Main Computer Program Flow Chart

```
WELCOME TO THE SINGULARLY PERTURBED ASYMPTOTIC METHOD
(SPAM FOR SHORT) FOR IRREGULAR PLANTS - VERSION 3.2
MAKE SURE BOTH THE IMSL AND CC6600 LIBRARIES ARE SET

    DESIGN 1 (1)  OTHER (2) :
   45000B   CM STORAGE USED
      1.812 CP SECONDS COMPILATION TIME
      NON-FATAL LOADER ERRORS -
 NON-EXISTENT LIBRARY GIVEN -  SYSIO   1

 THE A11 MATRIX IS -

      0.0000

 THE A12 MATRIX IS -

      0.0000          1.0000          .0318

 THE A21 MATRIX IS -

       .0489
      0.0000
      0.0000

 THE A22 MATRIX IS -

      -.3744           .0318         -.9995
    -39.9283        -2.1138           .0797
      6.2379         -.0063          -.7074

 THE B1 MATRIX IS A 1 BY 3 ZERO MATRIX

 THE B2 MATRIX IS -

       .0531           .0046          .0250
      9.7929        -55.7388         2.6033
     -5.0544         -1.8927         4.3880

  SYSTEM DIMENSION = 4
  # INPUTS/OUTPUTS = 3


  ENTER C1 ( 3 BY 1 )

 ROW 1 : 0
 ROW 2 : 1
 ROW 3 : 0

  ENTER C2 ( 3 BY 3 )

 ROW 1 : 1,0,0
 ROW 2 : 0,0,0
 ROW 3 : 0,0,1
```

```
THIS METHOD IS FOR IRREGULAR PLANTS, THAT IS, FOR
C2B2 NOT OF FULL RANK
*** TERMINAL ERROR            (IER = 129) FROM IMSL ROUTINE LUDATF
*** TERMINAL ERROR            (IER = 130) FROM IMSL ROUTINE LINV3F
THE PLANT IS IRREGULAR


ENTER THE M MATRIX ( 3 BY 1 ) :

 ROW 1 : 0
 ROW 2 : .25
 ROW 3 : 0


THE MATRIX F1 IS -

      0.0000
      1.0000
      0.0000

THE MATRIX F2 IS -

      1.0000          0.0000          0.0000
      0.0000           .2500           .0080
      0.0000          0.0000          1.0000

THE MATRIX F2*B2 IS -

       .0531           .0046           .0250
      2.4080        -13.9498           .5857
     -5.0544         -1.8927          4.3880

NOW VALUES OF SIGMA MUST BE CHOSEN. HOW DO YOU WANT
TO INPUT THESE (0=DIAGONAL, 1=FULL) : 0

ENTER THE 3 VALUES OF SIGMA (MUST BE POSITIVE) : 1,15,30

THE MATRIX K0 IS -

     11.6944           .1981         -2.0615
      2.7389         -1.0522          -.1395
     14.6517          -.2256          4.4020

THE MATRIX K1 MUST NOW BE INPUT. HOW DO YOU WISH
TO DO THIS (0=SCRATCH , 1=MULTIPLE OF K0) : 1

WHAT MULTIPLE OF K0 IS K1 : 2

 THE K1 MATRIX IS -

     23.3888           .3962         -4.1231
      5.4778         -2.1044          -.2789
     29.3035          -.4513          8.8041
```

```
HE FIRST SET OF SLOW EIGENVALUES ARE :

LAMBDA(1) = (-2.,0.)

LAMBDA(2) = (-2.,0.)

LAMBDA(3) = (-2.,0.)


THE SECOND SET OF SLOW EIGENVALUES ARE :

LAMBDA(4) = (-4.,0.)

THE FAST EIGENVALUES ARE (MUST BE MULTIPLIED BY G) :

LAMBDA(5) = (-1.,0.)

LAMBDA(6) = (-30.,0.)

LAMBDA(7) = (-15.,0.)

FORM CLOSED LOOP MATRICES (1), CHANGE OUTPUTS/EIGENVALUES
 (2),  OR STOP (0) : 1

PRINT C-L MATRICES (1=YES) : 0

ENTER THE GAIN CONSTANT , G : 29

ENTER THE 3 ACTUATOR CONTANTS :
 RUDDER CONSTANT : 20
AILERON CONSTANT : 20
 CANNARD CONSTANT : 20

THE OVERALL CLOSED LOOP EIGENVALUES ARE :

 EIGENVALUE (1) = (-9.348964874273,131.5128606026)
 EIGENVALUE (2) = (-9.348964874273,-131.5128606026)
 EIGENVALUE (3) = (-8.041546031385,92.59541875491)
 EIGENVALUE (4) = (-8.041546031385,-92.59541875491)
 EIGENVALUE (5) = (-9.135660521417,21.57045712481)
 EIGENVALUE (6) = (-9.135660521417,-21.57045712481)
 EIGENVALUE (7) = (-4.027317394378,0.)
 EIGENVALUE (8) = (-2.104785028134,0.)
 EIGENVALUE (9) = (-2.010762726775,0.)
 EIGENVALUE (10) = (-2.000431356545,0.)

CHANGE OUTPUTS/EIGENVALUES (1), CHANGE ONLY GAIN (2),
CHANGE SIGMA (K0/K1) (3), INTEGRATE C-L EQNS (4),
 CHANGE PLANT (5), STOP (0) : 4

 ******  ENTER INTEGRATING ROUTINE  ******
FOR THIS SYSTEM THERE ARE 3 INPUTS AND 3 OUTPUTS
DO YOU WISH TO SEE THE INPUT HISTORIES OR OUTPUT
 HISTORIES ? (1=INPUT,2=OUTPUT) : 2
```

```
ENTER MANEUVER (Y.P.=1,W.L.T.=2,H.T.=3) : 1

   ENTER INPUT COMMAND
   FINAL VALUE, RAMP TIME (UNIT STEP = 1,0)
1 : -3,1
2 : 0,0
  ENTER STEP RESPONSE END TIME (INFINITY=0) : 0
3 : 3,0
  ENTER STEP RESPONSE END TIME (INFINITY=0) : 1

ENTER FINAL TIME AND TIME STEP : 1.75,.05
```

T	OUTPUTS				
0.00	0.0000	0.0000	0.0000	0.0000	0.0000
.05	-.1936	.0014	1.1608	.0580	-.1356
.10	-.3831	.0028	1.9707	.1566	-.2266
.15	-.5369	.0044	2.4666	.2809	-.2560
.20	-.6728	.0062	2.7875	.4203	-.2525
.25	-.8110	.0079	2.9457	.5676	-.2434
.30	-.9563	.0095	3.0173	.7184	-.2379
.35	-1.1077	.0112	3.0411	.8705	-.2372
.40	-1.2591	.0129	3.0414	1.0226	-.2365
.45	-1.4072	.0144	3.0323	1.1742	-.2330
.50	-1.5562	.0159	3.0213	1.3252	-.2310
.55	-1.7034	.0174	3.0115	1.4758	-.2276
.60	-1.8586	.0188	3.0040	1.6260	-.2325
.65	-2.0098	.0201	2.9989	1.7759	-.2339
.70	-2.1550	.0212	2.9955	1.9257	-.2293
.75	-2.3016	.0223	2.9934	2.0754	-.2262
.80	-2.4463	.0233	2.9923	2.2250	-.2213
.85	-2.6002	.0242	2.9916	2.3746	-.2256
.90	-2.7491	.0251	2.9912	2.5242	-.2249
.95	-2.8976	.0259	2.9910	2.6737	-.2239
1.00	-3.0552	.0267	2.9909	2.8233	-.2319
1.05	-3.0155	.0261	1.7695	2.9117	-.1038
1.10	-2.9676	.0254	1.2151	2.9725	.0049
1.15	-2.9602	.0241	.7618	3.0106	.0504
1.20	-2.9779	.0229	.4358	3.0324	.0545
1.25	-2.9932	.0217	.2229	3.0435	.0503
1.30	-2.9964	.0205	.0954	3.0483	.0519
1.35	-2.9930	.0192	.0261	3.0496	.0566
1.40	-2.9903	.0178	-.0069	3.0492	.0589
1.45	-2.9907	.0166	-.0193	3.0483	.0576
1.50	-2.9925	.0153	-.0211	3.0472	.0547
1.55	-2.9940	.0142	-.0183	3.0463	.0523
1.60	-2.9946	.0131	-.0142	3.0456	.0510
1.65	-2.9949	.0120	-.0104	3.0451	.0502
1.70	-2.9952	.0110	-.0073	3.0447	.0495
1.75	-2.9957	.0101	-.0051	3.0444	.0488

```
BETA COMMAND =-1.633031846067
R COMMAND =2.823251138512
PHI =.02907321598383
BETA TRANS. =.01173669636142
R TRANS. =.3810131352851
```

```
HANGE OUTPUTS/EIGENVALUES (1), CHANGE ONLY GAIN (2),
CHANGE SIGMA (K0/K1) (3), INTEGRATE C-L EQNS (4),
 CHANGE PLANT (5), STOP (0) : 5

    NEW DESIGN (1) OR ROBUSTNESS TEST (2) ? : 2

ENTER THE DIMENSION OF THE SYSTEM : 4
ENTER THE NUMBER OF ACTIVE INPUTS (SIZE OF B2) : 3
  ENTER A11 ( 1 BY 1 )

 ROW 1 : 0

  ENTER A12 ( 1 BY 3 )

 ROW 1 : 0,1,.0255

  ENTER A21 ( 3 BY 1 )

 ROW 1 : .0332
 ROW 2 : 0
 ROW 3 : 0

  ENTER A22 ( 3 BY 3 )

 ROW 1 : -.4836,.0255,-.9997
 ROW 2 : -75.3125,-3.9178,.0382
 ROW 3 : 10.6995,-.0294,-.5117

B1 IS A ZERO MATRIX OF DIMENSION 1 BY 3

  ENTER B2 ( 3 BY 3 )

 ROW 1 : .0473,.0066,.0399
 ROW 2 : 13.4743,-82.0763,8.3887
 ROW 3 : -7.7774,-3.2921,8.7542
*** TERMINAL ERROR          (IER = 129) FROM IMSL ROUTINE LUDATF
*** TERMINAL ERROR          (IER = 130) FROM IMSL ROUTINE LINV3F
THE PLANT IS IRREGULAR


THE MATRIX F1 IS -

      0.0000
      1.0000
      0.0000

THE MATRIX F2 IS -

      1.0000         0.0000         0.0000
      0.0000          .2500          .0064
      0.0000         0.0000         1.0000
```

HE MATRIX F2*B2 IS -

.0473	.0066	.0399
3.3190	-20.5401	2.1530
-7.7774	-3.2921	8.7542

THE MATRIX KO IS -

11.6944	.1981	-2.0615
2.7389	-1.0522	-.1395
14.6517	-.2256	4.4020

THE K1 MATRIX IS -

23.3888	.3962	-4.1231
5.4778	-2.1044	-.2789
29.3035	-.4513	8.8041

THE FIRST SET OF SLOW EIGENVALUES ARE :

LAMBDA(1) = (-2.,0.)

LAMBDA(2) = (-2.,0.)

LAMBDA(3) = (-2.,0.)

THE SECOND SET OF SLOW EIGENVALUES ARE :

LAMBDA(4) = (-4.,0.)

THE FAST EIGENVALUES ARE (MUST BE MULTIPLIED BY G) :

LAMBDA(5) = (-1.11882838742,0.)

LAMBDA(6) = (-55.06024604491,0.)

LAMBDA(7) = (-21.78931709532,0.)

THE OVERALL CLOSED LOOP EIGENVALUES ARE :

EIGENVALUE (1) = (-9.511823000371,178.4317581339)
EIGENVALUE (2) = (-9.511823000371,-178.4317581339)
EIGENVALUE (3) = (-8.918649595212,111.9021828101)
EIGENVALUE (4) = (-8.918649595212,-111.9021828101)
EIGENVALUE (5) = (-8.963727600824,23.04292922593)
EIGENVALUE (6) = (-8.963727600824,-23.04292922593)
EIGENVALUE (7) = (-3.951230438002,0.)
EIGENVALUE (8) = (-2.121727793373,0.)
EIGENVALUE (9) = (-2.005970702296,.002953256932167)
EIGENVALUE (10) = (-2.005970702296,-.002953256932167)

**

```
NTER INTEGRATING ROUTINE  ******
FOR THIS SYSTEM THERE ARE 3 INPUTS AND 3 OUTPUTS
DO YOU WISH TO SEE THE INPUT HISTORIES OR OUTPUT
 HISTORIES ? (1=INPUT,2=OUTPUT) : 2
 ENTER MANEUVER (Y.P.=1,W.L.T.=2,H.T.=3) : 1

    ENTER INPUT COMMAND
    FINAL VALUE, RAMP TIME (UNIT STEP = 1,0)
1 : -3,.75
2 : 0,0
   ENTER STEP RESPONSE END TIME (INFINITY=0) : 0
3 : 4,0
   ENTER STEP RESPONSE END TIME (INFINITY=0) : .75

ENTER FINAL TIME AND TIME STEP : 1.5,.05
```

T	OUTPUTS				
0.00	0.0000	0.0000	0.0000	0.0000	0.0000
.05	-.2520	-.0216	6.0860	.3043	.0523
.10	-.5291	-.0078	3.2783	.4682	-.0609
.15	-.7149	-.0059	4.0252	.6695	-.0454
.20	-.8935	.0076	4.2972	.8843	-.0091
.25	-1.0717	.0090	3.6887	1.0688	-.0029
.30	-1.2708	.0118	4.2317	1.2804	.0096
.35	-1.4809	.0139	3.8643	1.4736	-.0074
.40	-1.6837	.0157	4.0574	1.6764	-.0073
.45	-1.8823	.0184	3.9882	1.8759	-.0065
.50	-2.0740	.0201	3.9926	2.0755	.0015
.55	-2.2712	.0218	4.0049	2.2757	.0045
.60	-2.4731	.0234	3.9821	2.4748	.0017
.65	-2.6779	.0253	3.9960	2.6746	-.0033
.70	-2.8755	.0273	3.9971	2.8745	-.0010
.75	-3.0723	.0297	3.9862	3.0738	.0015
.80	-3.0254	.0543	-1.8588	2.9809	-.0446
.85	-2.9424	.0390	.4810	3.0049	.0625
.90	-2.9556	.0379	.1216	3.0110	.0554
.95	-2.9753	.0255	-.3879	2.9916	.0163
1.00	-2.9935	.0247	.3308	3.0081	.0147
1.05	-2.9927	.0231	-.2394	2.9962	.0034
1.10	-2.9859	.0216	.1051	3.0014	.0155
1.15	-2.9858	.0209	-.0525	2.9988	.0130
1.20	-2.9877	.0190	-.0132	2.9981	.0104
1.25	-2.9911	.0180	.0088	2.9986	.0075
1.30	-2.9922	.0166	.0263	2.9973	.0050
1.35	-2.9925	.0154	.0059	2.9976	.0050
1.40	-2.9929	.0141	-.0141	2.9968	.0039
1.45	-2.9936	.0128	-.0033	2.9967	.0031
1.50	-2.9944	.0118	-.0058	2.9964	.0019

```
BETA COMMAND =-1.261148520343
R COMMAND =3.073795857869
PHI =.03064685696701
BETA TRANS. =.01610555728732
R TRANS. =.3820396586508
```

```
      PROGRAM PORTER(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
     1 PLOT,TAPE4=PLOT)
C
      REAL A11(10,10),A12(10,10),A21(10,10),A22(10,10)
      REAL B1(10,10),B2(10,10),C1(10,10),C2(10,10)
      REAL F1(10,10),F2(10,10),M(10,10),UC(10)
      REAL K0(10,10),K1(10,10),X(10),Y(10),Z(10)
      REAL A12FF(10,10),V(10),C2B2(10,10)
      REAL WKAREA(10),MA11(10,10),MA12(10,10),F2B2(10,10)
      REAL SIGMA(10,10),FBINV(10,10),K00(10,10),K11(10,10)
      REAL BETA(10),SLOW2(10),AFIF(10,10),F2IF1(10,10),F2INV(10,10)
      COMPLEX ALFA(10),LAMBDA(10),LAM(10),W(10),W1(10),ZQ(10,10)
      REAL AY(10,10),BEE(10,10),CEE(10,10)
      REAL AC(10,10),ACK1(10,10),ACKF1(10,10),ACKF2(10,10),ACK0(10,10)
      REAL FB2(10,10),F2B2K0(10,10),FAST(10,10)
      REAL IDEN(10,10),F2S(10,10),AY1(10,10)
      REAL XP(10),MULT,ZDOT(10),U(10),ER(5),RR(2)
      REAL VC(10),VDT(10),SLOPE(10),VST(10)
      INTEGER KVR(10),KVST(10)
      DIMENSION WORK(500),IWORK(10)
      COMMON AY,BEE,V,NL,L
      EXTERNAL F
C
C
      DO 99 I=1,3
   99 PRINT*," "
C
      PRINT*,"WELCOME TO THE SINGULARLY PERTURBED ASYMPTOTIC METHOD"
      PRINT*,"(SPAM FOR SHORT) FOR IRREGULAR PLANTS - VERSION 3.2"
      PRINT*,"MAKE SURE BOTH THE IMSL AND CC6600 LIBRARIES ARE SET"
      PRINT*," "
C
      X10=0
      X11=1
      PRINT*,"    DESIGN 1 (1)  OTHER (2) :"
      READ*,XX
      IF(XX.EQ.1) GOTO 10
C
 9970 IF(X10.NE.5) GO TO 9971
      PRINT*," "
      PRINT*,"    NEW DESIGN (1) OR ROBUSTNESS TEST (2) ? :"
      READ*,X11
 9971 CONTINUE
      PRINT*," "
      PRINT*,"ENTER THE DIMENSION OF THE SYSTEM : "
      READ*,N
C
      IF (N.LE.7) GOTO 100
      PRINT*,"THE DIMENSIONS MUST BE CHANGED TO HANDLE THAT"
      GOTO 9999
C
  100 PRINT*,"ENTER THE NUMBER OF ACTIVE INPUTS (SIZE OF B2) : "
      READ*,L
      NM=N-L
```

```
C
      PRINT*,"  ENTER A11 ( ",NM," BY ",NM," )"
      CALL READM(NM,NM,A11)
      PRINT*," "
      PRINT*,"  ENTER A12 ( ",NM," BY ",L," )"
      CALL READM(NM,L,A12)
      PRINT*," "
      PRINT*,"  ENTER A21 ( ",L," BY ",NM," )"
      CALL READM(L,NM,A21)
      PRINT*," "
      PRINT*,"  ENTER A22 ( ",L," BY ",L," )"
      CALL READM(L,L,A22)
C
      PRINT*," "
      PRINT*,"B1 IS A ZERO MATRIX OF DIMENSION ",NM," BY ",L
C
      DO 200 I=1,NM
      DO 200 J=1,L
  200 B1(I,J)=0.
C
      PRINT*," "
      PRINT*,"  ENTER B2 ( ",L," BY ",L," )"
      CALL READM(L,L,B2)
C
      IF(X11.EQ.2) GO TO 9972
C
      GOTO 20
C
   10 CALL DSGN1(N,L,A11,A12,A21,A22,B1,B2)
      PRINT*," "
      PRINT*,"  SYSTEM DIMENSION = ",N
      PRINT*,"  # INPUTS/OUTPUTS = ",L
      PRINT*," "
      NM=N-L
      GOTO 20
C
   30 PRINT*," "
      PRINT*,"NEW OUTPUT MATRICES ?(1=YES) :"
      READ*,X5
      IF (X5.NE.1) GOTO 380
C
   20 PRINT*," "
      PRINT*,"  ENTER C1 ( ",L," BY ",NM," )"
      CALL READM(L,NM,C1)
      PRINT*," "
      PRINT*,"  ENTER C2 ( ",L," BY ",L," )"
      CALL READM(L,L,C2)
C
      PRINT*," "
      PRINT*,"THIS METHOD IS FOR IRREGULAR PLANTS, THAT IS, FOR"
      PRINT*,"C2B2 NOT OF FULL RANK"
 9972 CALL VMULFF(C2,B2,L,L,L,10,10,C2B2,10,IER)
      D1=1.
      CALL LINV3F(C2B2,B,4,L,10,D1,D2,WKAREA,IER)
```

```
      DET=D1*2**D2
      IF (DET.EQ.0) GOTO 280
      PRINT*,"THIS PLANT IS REGULAR - PROGRAM TERMINATED"
      GOTO 9999
  280 PRINT*,"THE PLANT IS IRREGULAR"
      PRINT*," "
      IF(X11.EQ.2) GO TO 9973
C
  380 CONTINUE
  610 PRINT*," "
      PRINT*," "
      PRINT*,"ENTER THE M MATRIX ( ",L," BY ",NM," ) : "
      CALL READM(L,NM,M)
      PRINT*," "
 9973 CALL VMULFF(M,A11,L,NM,NM,10,10,MA11,10,IER)
      CALL MADD(C1,MA11,L,NM,F1)
      CALL VMULFF(M,A12,L,NM,L,10,10,MA12,10,IER)
      CALL MADD(C2,MA12,L,L,F2)
      CALL MADD(C2,MA12,L,L,F2S)
      CALL VMULFF(F2,B2,L,L,L,10,10,F2B2,10,IER)
      PRINT*," "
      PRINT*,"THE MATRIX F1 IS -"
      CALL PRINTM(L,NM,F1)
      PRINT*," "
      PRINT*,"THE MATRIX F2 IS -"
      CALL PRINTM(L,L,F2)
      PRINT*," "
      PRINT*,"THE MATRIX F2*B2 IS -"
      CALL PRINTM(L,L,F2B2)
      D1=1
      CALL LINV3F(F2B2,B,4,L,10,D1,D2,WKAREA,IER)
      DET=D1*2**D2
      IF (DET.NE.0) GOTO 370
      PRINT*,"THE MATRIX F2*B2 IS NOT FULL RANK - TRY AGAIN"
      GOTO 30
  370 CONTINUE
      IF(X11.EQ.2) GO TO 9974
      PRINT*," "
      PRINT*,"NOW VALUES OF SIGMA MUST BE CHOSEN. HOW DO YOU WANT"
      PRINT*,"TO INPUT THESE (0=DIAGONAL, 1=FULL) :"
      READ*,JJ
      IF (JJ.EQ.1) GOTO 390
C
      DO 393 I=1,L
      DO 393 J=1,L
  393 SIGMA(I,J)=0
C
      PRINT*," "
      PRINT*,"ENTER THE ",L," VALUES OF SIGMA (MUST BE POSITIVE) :"
C
      DO 400 I=1,L
  400 READ*,SIGMA(I,I)
C
      GOTO 410
```

```
C
  390 PRINT*," "
      PRINT*,"ENTER SIGMA -"
      CALL READM(L,L,SIGMA)
C
 9974 CONTINUE
  410 CALL VMULFF(F2,B2,L,L,L,10,10,F2B2,10,IER)
C
      DO 392 I=1,L
      DO 392 J=1,L
  392 FB2(I,J)=F2B2(I,J)
C
      IF(X11.EQ.2) GO TO 9975
      CALL LINV1F(F2B2,L,10,FBINV,10,WKAREA,IER)
      CALL VMULFF(FBINV,SIGMA,L,L,L,10,10,K0,10,IER)
C
 9975 PRINT*," "
      PRINT*,"THE MATRIX K0 IS -"
      CALL PRINTM(L,L,K0)
C
      DO 460 I=1,L
      DO 460 J=1,L
  460 K00(I,J)=-K0(I,J)
C
      IF(X11.EQ.2) GO TO 9976
      PRINT*," "
      PRINT*,"THE MATRIX K1 MUST NOW BE INPUT. HOW DO YOU WISH"
      PRINT*,"TO DO THIS (0=SCRATCH , 1=MULTIPLE OF K0) :"
      READ*,G2
 9976 IF (G2.EQ.1) GO TO 480
  520 CONTINUE
      PRINT*," "
      PRINT*,"  ENTER K1 (",L," BY ",L,")"
      CALL READM(L,L,K1)
C
      GOTO 510
  480 IF (X11.EQ.2) GO TO 9977
      PRINT*," "
      PRINT*,"WHAT MULTIPLE OF K0 IS K1 :"
      READ*,MULT
C
      DO 522 I=1,L
      DO 522 J=1,L
  522 K1(I,J)=MULT*K0(I,J)
C
 9977 PRINT*," "
      PRINT*," THE K1 MATRIX IS -"
      CALL PRINTM(L,L,K1)
C
      DO 523 I=1,L
      DO 523 J=1,L
  523 K11(I,J)=K1(I,J)
C
  510 CALL EIGZF(K11,10,K00,10,L,0,ALFA,BETA,ZQ,10,WKAREA,IER)
```

```
C
      DO 530 I=1,L
      YY=BETA(I)
      IF (YY.EQ.0) GOTO 540
      LAMBDA(I)=ALFA(I)/BETA(I)
      GOTO 530
  540 LAMBDA(I)=9.E+99
  530 CONTINUE
C
      PRINT*," "
      PRINT*,"THE FIRST SET OF SLOW EIGENVALUES ARE :"
      PRINT*," "
C
      DO 550 I=1,L
      PRINT*,"LAMBDA(",I,") = ",LAMBDA(I)
  550 PRINT*," "
C
      CALL LINV1F(F2S,L,10,F2INV,10,WKAREA,IER)
      CALL VMULFF(F2INV,F1,L,L,NM,10,10,F2IF1,10,IER)
      CALL VMULFF(A12,F2IF1,NM,L,NM,10,10,A12FF,10,IER)
C
      DO 560 I=1,NM
      DO 560 J=1,NM
  560 AFIF(I,J)=-A12FF(I,J)
C
      CALL MADD(A11,AFIF,NM,NM,SLOW2)
      CALL EIGRF(SLOW2,NM,10,0,W,ZQ,10,WKAREA,IER)
C
      PRINT*," "
      PRINT*,"THE SECOND SET OF SLOW EIGENVALUES ARE :"
      PRINT*," "
C
      DO 580 I=1,NM
      J=I+L
      PRINT*,"LAMBDA(",J,") = ",W(I)
  580 PRINT*," "
C
      CALL VMULFF(FB2,K0,L,L,L,10,10,F2B2K0,10,IER)
C
      DO 524 I=1,L
      DO 524 J=1,L
  524 FAST(I,J)=-F2B2K0(I,J)
C
      CALL EIGRF(FAST,L,10,0,W1,ZQ,10,WKAREA,IER)
C
      PRINT*,"THE FAST EIGENVALUES ARE (MUST BE MULTIPLIED BY G) :"
      PRINT*," "
      DO 590 I=1,L
      J=NM+L+I
      PRINT*,"LAMBDA(",J,") = ",W1(I)
  590 PRINT*," "
      IF(X11.EQ.2) GO TO 9978
C
      PRINT*,"FORM CLOSED LOOP MATRICES (1), CHANGE OUTPUTS/EIGENVALUES"
```

```
      PRINT*," (2),  OR STOP (0) :"
      READ*,X3
      IF (X3.EQ.2) GOTO 30
      IF (X3.EQ.0) GOTO 9999
 3999 PRINT*," "
      PRINT*,"PRINT C-L MATRICES (1=YES) :"
      READ*,X25
      PRINT*," "
      PRINT*,"ENTER THE GAIN CONSTANT , G :"
      READ*,G
C
      DO 325 I=1,L
      DO 325 J=1,L
  325 AC(I,J)=0
C
      PRINT*," "
      PRINT*,"ENTER THE ",L," ACTUATOR CONSTANTS :"
      PRINT*,"RUDDER CONSTANT :"
      READ*,AC(1,1)
      PRINT*,"AILERON CONSTANT :"
      READ*,AC(2,2)
      PRINT*,"CANNARD CONSTANT :"
      READ*,AC(3,3)
C
 9978 CONTINUE
      I1=N+L+L
      I2=L+NM
      I3=L+L+NM
C
      DO 600 I=1,I1
      DO 620 J=1,I1
  620 AY(I,J)=0.0
      DO 630 K=1,L
      BEE(I,K)=0.0
  630 CEE(K,I)=0.0
  600 CONTINUE
C
      CALL VMULFF(K0,F2,L,L,L,10,10,K0F2,10,IER)
      CALL VMULFF(AC,K1,L,L,L,10,10,ACK1,10,IER)
      CALL VMULFF(AC,K0F2,L,L,L,10,10,ACKF2,10,IER)
      CALL VMULFF(AC,K0,L,L,L,10,10,ACK0,10,IER)
      CALL VMULFF(ACK0,F1,L,L,NM,10,10,ACKF1,10,IER)
C
      DO 640 I=1,NM
      DO 640 J=1,NM
  640 AY(L+I,L+J)=A11(I,J)
C
      DO 650 I=1,L
      DO 650 J=1,NM
      AY(I,L+J)=-F1(I,J)
      AY(I2+I,L+J)=A21(I,J)
      AY(I3+I,L+J)=-G*ACKF1(I,J)
      AY(L+J,I2+I)=A12(J,I)
  650 CEE(I,L+J)=C1(I,J)
```

```
C
      DO 660 I=1,L
      DO 660 J=1,L
      AY(I,I2+J)=-F2(I,J)
      AY(I2+I,I2+J)=A22(I,J)
      AY(I2+I,I3+J)=B2(I,J)
      AY(I3+I,J)=G*ACK1(I,J)
      AY(I3+I,I2+J)=-G*ACKF2(I,J)
      AY(I3+I,I3+I)=-AC(I,I)
      CEE(I,I2+J)=C2(I,J)
      BEE(I,I)=1.0
  660 BEE(I3+I,J)=G*ACK0(I,J)
C
      NL=I1
      IF (X25.NE.1) GOTO 4310
      PRINT*," "
      PRINT*,"THE CLOSED LOOP AY MATRIX IS -"
      CALL PRINTM(NL,NL,AY)
      PRINT*," "
      PRINT*,"THE CLOSED LOOP BEE MATRIX IS -"
      CALL PRINTM(NL,L,BEE)
      PRINT*," "
      PRINT*,"THE CLOSED LOOP CEE MATRIX IS -"
      CALL PRINTM(L,NL,CEE)
 4310 CONTINUE
      PRINT*," "
C
      DO 4620 I=1,NL
      DO 4620 J=1,NL
 4620 AY1(I,J)=AY(I,J)
C
      PRINT*,"THE OVERALL CLOSED LOOP EIGENVALUES ARE :"
      PRINT*," "
      CALL EIGRF(AY1,NL,10,0,W1,ZQ,10,WKAREA,IER)
C
      DO 4700 I=1,NL
 4700 PRINT*," EIGENVALUE (",I,") = ",W1(I)
      IF(X11.EQ.2) GO TO 9979
C
 4710 CONTINUE
      PRINT*," "
      PRINT*,"CHANGE OUTPUTS/EIGENVALUES (1), CHANGE ONLY GAIN (2),"
      PRINT*,"CHANGE SIGMA (K0/K1) (3), INTEGRATE C-L EQNS (4),"
      PRINT*," CHANGE PLANT (5), STOP (0) :"
      READ*,X10
      IF (X10.EQ.0) GOTO 9999
      IF (X10.EQ.1) GOTO 30
      IF (X10.EQ.2) GOTO 3999
      IF (X10.EQ.3) GOTO 370
      IF (X10.EQ.5) GOTO 9970
C
C
 9979 PRINT*," "
      PRINT*," ******  ENTER INTEGRATING ROUTINE  ******"
```

```
      PRINT*,"FOR THIS SYSTEM THERE ARE ",L," INPUTS AND ",L," OUTPUTS"
      PRINT*,"DO YOU WISH TO SEE THE INPUT HISTORIES OR OUTPUT"
      PRINT*," HISTORIES ? (1=INPUT,2=OUTPUT) :"
      READ*,AA1
      PRINT*," ENTER MANEUVER (Y.P.=1,W.L.T.=2,H.T.=3) :"
      READ*,AA2
      PRINT*," "
      PRINT*,"    ENTER INPUT COMMAND"
      PRINT*,"    FINAL VALUE, RAMP TIME (UNIT STEP = 1,0)"
C
      DO 1111 I=1,L
      KVR(I)=0
      SLOPE(I)=0.0
      VST(I)=0.0
      KVST(I)=0
      PRINT*,I," : "
      READ*,VC(I),VDT(I)
      VC(I)=VC(I)/57.2957795
      IF(VDT(I).EQ.0) GO TO 1160
      KVR(I)=1
      SLOPE(I)=VC(I)/VDT(I)
      GO TO 1111
 1160 PRINT*,"   ENTER STEP RESPONSE END TIME (INFINITY=0) :"
      READ*,VST(I)
      IF(VST(I).EQ.0) GO TO 1165
      KVST(I)=1
 1165 V(I)=VC(I)
 1111 CONTINUE
      TDT=VDT(1)
      IF(AA2.EQ.2) TDT=VST(3)
      FV=VC(1)*57.2957795
C
      PRINT*," "
      PRINT*,"ENTER FINAL TIME AND TIME STEP :"
      READ*,TF,DELT
      PRINT*," "
      NEQN=NL
      T=0
      TOUT=0
C
      DO 1113 I=1,NL
      X(I)=0
      Y(I)=0
 1113 U(I)=0
C
      DO 1112 I=1,5
 1112 ER(I)=0
C
      RR(1)=0
      RR(2)=0
      RELERR=1E-06
      ABSERR=1E-06
      IFLAG=1
      NSTEP=INT(TF/DELT)+1
```

```
C
      IF(AA1.EQ.1) GO TO 131
      PRINT*,"        T         OUTPUTS"
      GO TO 132
  131 PRINT*,"       T         CONTROLS"
  132 PRINT*,"-----------------------------------------------------"
      IF(AA1.EQ.1) GO TO 125
      IF(AA2.EQ.1) GO TO 126
  125 PRINT 9980,TOUT,(U(I),I=1,L)
      WRITE(4,9980) TOUT,(U(I),I=1,L)
      GO TO 127
  126 PRINT 9981,TOUT,(Y(I),I=1,L),RR(1),RR(2)
      WRITE(4,9981) TOUT,(Y(I),I=1,L),RR(1),RR(2)
  127 CONTINUE
C
C
      DO 1115 K=1,NSTEP
C
       TOUT=TOUT+DELT
C
        DO 1150 I=1,L
        IF(KVR(I).EQ.0) GO TO 1155
        V(I)=SLOPE(I)*TOUT
        IF(TOUT.GT.VDT(I)) V(I)=VC(I)
        GO TO 1150
 1155   IF(KVST(I).EQ.0) GO TO 1150
        IF(TOUT.GT.VST(I)) V(I)=0.0
 1150   CONTINUE
C
       CALL ODE(F,NEQN,X,T,TOUT,RELERR,ABSERR,IFLAG,WORK,IWORK)
C
      IF(AA1.EQ.2) GO TO 133
C
      DO 1120 I=1,L
 1120 U(I)=X(I3+I)
C
C
C
        DO 1136 I=1,L
 1136   UC(I)=U(I)*57.29578
C
C
       PRINT 9980,TOUT,(UC(I),I=1,L)
       WRITE(4,9980) TOUT,(UC(I),I=1,L)
      GO TO 134
C
  133 DO 1128 I=1,L
 1128 Y(I)=0
C
      DO 135 I=1,L
      DO 135 J=1,NL
  135 Y(I)=Y(I)+CEE(I,J)*X(J)
C
      DO 136 I=1,L
  136 Y(I)=Y(I)*57.2957795
```

```
C
      KK=2
      IF(TOUT.EQ.TDT) KK=1
      IF(AA2.EQ.1) GO TO 137
      IF(AA2.EQ.2) GO TO 139
C
      IF(TOUT.GT.TDT) GO TO 141
      ER(1)=ER(1)+Y(1)*DELT*KK
      ER(2)=0
      ER(4)=0
  141 ER(3)=ER(3)+ABS(Y(2)*DELT*KK)
      E4=(Y(1)-FV)*DELT*KK
      ER(4)=ER(4)+ABS(E4)
      ER(5)=ER(5)+ABS(Y(3)*DELT*KK)
      GO TO 142
C
  137 IF(TOUT.GT.TDT) GO TO 138
      ER(1)=ER(1)+Y(1)*DELT*KK
      ER(2)=ER(2)+Y(3)*DELT*KK
      ER(4)=0
      ER(5)=0
  138 ER(3)=ER(3)+ABS(Y(2)*DELT*KK)
      E4=(Y(1)-FV)*DELT*KK
      ER(4)=ER(4)+ABS(E4)
      ER(5)=ER(5)+ABS(Y(3)*DELT*KK)
      RR(1)=RR(1)+Y(3)*DELT
      RR(2)=RR(1)+Y(1)
      GO TO 142
C
  139 IF(TOUT.GT.TDT) GO TO 140
      ER(1)=0
      ER(2)=ER(2)+Y(3)*DELT*KK
      ER(5)=0
  140 ER(3)=ER(3)+ABS(Y(2)*DELT*KK)
      ER(4)=ER(4)+ABS(Y(1)*DELT*KK)
      ER(5)=ER(5)+ABS(Y(3)*DELT*KK)
C
  142 IF(AA2.EQ.1) GO TO 128
      PRINT 9980,TOUT,(Y(I),I=1,L)
      WRITE(4,9980) TOUT,(Y(I),I=1,L)
      GO TO 129
  128 PRINT 9981,TOUT,(Y(I),I=1,L),RR(1),RR(2)
      WRITE(4,9981) TOUT,(Y(I),I=1,L),RR(1),RR(2)
  129 CONTINUE
C
  134 CONTINUE
 1115 CONTINUE
 9980 FORMAT(5X,F5.2,3(3X,F10.4))
 9981 FORMAT(5X,F5.2,3(3X,F10.4),2X,F7.4,2X,F7.4)
      PRINT*,"---------------------------------------------------"
      PRINT*," "
      IF(AA1.EQ.1) GO TO 146
      DO 147 I=1,5
  147 ER(I)=ER(I)/KK
C
      PRINT*," BETA COMMAND =",ER(1)
```

```
      PRINT*," R COMMAND =",ER(2)
      PRINT*," PHI =",ER(3)
      PRINT*," BETA TRANS. =",ER(4)
      PRINT*," R TRANS. =",ER(5)
      PRINT*," "
  146 CONTINUE
      GOTO 4710
C
C
C
 9999 CONTINUE
      PRINT*," "
      PRINT*,"ALL INFORMATION IN PORTER HAS BEEN SAVED IN LOCAL"
      PRINT*,"FILE - PLOT"
      STOP
      END
C
C
C
      SUBROUTINE MADD(A,B,L,N,C)
      DIMENSION A(10,10),B(10,10),C(10,10)
      DO 100 I=1,L
      DO 100 J=1,N
  100 C(I,J)=A(I,J)+B(I,J)
      RETURN
      END
C
      SUBROUTINE READM(L,N,A)
      REAL A(10,10)
      PRINT*," "
      DO 100 I=1,L
      PRINT*," ROW ",I," :"
  100 READ*,(A(I,J),J=1,N)
      RETURN
      END
C
      SUBROUTINE PRINTM(L,N,A)
      REAL A(10,10)
      PRINT*," "
      DO 100 I=1,L
  100 PRINT 200,(A(I,J),J=1,N)
  200 FORMAT(1X,8(F12.4,3X))
      RETURN
      END
C
C
      SUBROUTINE MSUB(A,B,C,L,N)
      REAL A(10,10),B(10,10),C(10,10)
      DO 100 I=1,L
      DO 100 J=1,N
  100 C(I,J)=A(I,J)-B(I,J)
      RETURN
      END
C
C
      SUBROUTINE DSGN1(N,L,A,B,C,D,E,F)
```

```
      REAL A(10,10),B(10,10),C(10,10),D(10,10),E(10,10),F(10,10)
      N=4
      L=3
      A(1,1)=0.
      B(1,1)=0.0  $B(1,2)=1.0  $B(1,3)=.03184253
      C(1,1)=.04885258  $C(2,1)=0.0   $C(3,1)=0.
      D(1,1)=-.37442438  $D(1,2)=.03182685  $D(1,3)=-.99948763
      D(2,1)=-39.92833732  $D(2,2)=-2.11380581  $D(2,3)=.07969577
      D(3,1)=6.23794780  $D(3,2)=-.00631591  $D(3,3)=-.70740917
      DO 100 I=1,3
  100 E(1,I)=0.0
      F(1,1)=.05309116  $F(1,2)=.00463283  $F(1,3)=.02501018
      F(2,1)=9.79285579  $F(2,2)=-55.73875487  $F(2,3)=2.60326273
      F(3,1)=-5.05439681  $F(3,2)=-1.89267784  $F(3,3)=4.38800282
      PRINT*," "
      PRINT*," THE A11 MATRIX IS -"
      CALL PRINTM(1,1,A)
      PRINT*," "
      PRINT*," THE A12 MATRIX IS -"
      CALL PRINTM(1,3,B)
      PRINT*," "
      PRINT*," THE A21 MATRIX IS -"
      CALL PRINTM(3,1,C)
      PRINT*," "
      PRINT*," THE A22 MATRIX IS -"
      CALL PRINTM(3,3,D)
      PRINT*," "
      PRINT*," THE B1 MATRIX IS A 1 BY 3 ZERO MATRIX "
      PRINT*," "
      PRINT*," THE B2 MATRIX IS -"
      CALL PRINTM(3,3,F)
      RETURN
      END
C
C
      SUBROUTINE F(T,X,XP)
      REAL X(10),XP(10),AY(10,10),BEE(10,10),V(10)
      COMMON AY,BEE,V,NL,L
C
      DO 100 I=1,NL
  100 XP(I)=0
C
      DO 200 I=1,NL
      DO 200 J=1,NL
  200 XP(I)=XP(I)+AY(I,J)*X(J)
C
      DO 300 I=1,NL
      DO 300 J=1,L
  300 XP(I)=XP(I)+BEE(I,J)*V(J)
C
      RETURN
      END
C
C
*EOR
*EOF
```

```
 3999 PRINT*," "
      PRINT*,"PRINT C-L MATRICES (1=YES) :"
      READ*,X25
      PRINT*," "
      PRINT*,"ENTER THE GAIN CONSTANT , G :"
      READ*,G
C
C        FORM CLOSED LOOP A-MATRIX (AY)
C
 9977 DO 4000 I=1,N
      DO 4000 J=1,L
 4000 AY(I,J)=0.
C
      CALL VMULFF(B2,K1,L,L,L,8,8,B2K1,8,IER)
      CALL VMULFF(B2,K0,L,L,L,8,8,B2K0,8,IER)
      CALL VMULFF(B2K0,F1,L,L,NM,8,8,B2K0F1,8,IER)
      CALL VMULFF(B2K0,F2,L,L,L,8,8,B2K0F2,8,IER)
C
      DO 4100 I=1,L
C
        DO 4090 J=1,L
        BKF2(I,J)=G*B2K0F2(I,J)
 4090   BK1(I,J)=G*B2K1(I,J)
C
        DO 4095 KA=1,NM
 4095   BKF1(I,KA)=G*B2K0F1(I,KA)
C
 4100 CONTINUE
C
      CALL MSUB(A21,BKF1,THRTWO,L,NM)
      CALL MSUB(A22,BKF2,THRTHR,L,L)
C
      DO 4200 I=1,L
C
        DO 4190 J=1,L
        KA=I+N
        LA=J+N
        AY(KA,J)=BK1(I,J)
        AY(I,LA)=-F2(I,J)
 4190   AY(KA,LA)=THRTHR(I,J)
C
        DO 4195 J=1,NM
        LB=J+L
        AY(I,LB)=-F1(I,J)
 4195   AY(KA,LB)=THRTWO(I,J)
C
 4200 CONTINUE
C
      DO 4300 I=1,NM
C
        DO 4290 J=1,L
        LA=I+L
        KA=J+N
 4290   AY(LA,KA)=A12(I,J)
C
```

```
              DO 4295 J=1,NM
              KB=J+L
      4295    AY(LA,KB)=A11(I,J)
C
      4300 CONTINUE
C
           NL=N+L
           IF (X25.NE.1) GOTO 4310
           PRINT*," "
           PRINT*,"THE CLOSED LOOP AY MATRIX IS -"
           CALL PRINTM(NL,NL,AY)
      4310 CONTINUE
C
C               FORM CLOSED LOOP B-MATRIX (BEE)
C
           DO 4400 I=1,NL
           DO 4400 J=1,L
      4400 BEE(I,J)=0.0
C
           DO 4410 I=1,L
      4410 BEE(I,I)=1.0
C
           DO 4420 I=1,L
           DO 4420 J=1,L
           GB2KO(I,J)=G*B2KO(I,J)
      4420 BEE(N+I,J)=GB2KO(I,J)
C
           IF (X25.NE.1) GOTO 4510
           PRINT*," "
           PRINT*,"THE CLOSED LOOP BEE MATRIX IS -"
           CALL PRINTM(NL,L,BEE)
      4510 CONTINUE
C
C               FORM CLOSED LOOP C-MATRIX (CEE)
C
           DO 4500 I=1,L
           DO 4500 J=1,NL
      4500 CEE(I,J)=0.0
C
           DO 4515 I=1,L
C
              DO 4520 J=1,NM
      4520    CEE(I,L+J)=C1(I,J)
C
              DO 4530 J=1,L
      4530    CEE(I,N+J)=C2(I,J)
C
      4515 CONTINUE
C
           PRINT*," "
           IF (X25.NE.1) GOTO 4610
           PRINT*,"THE CLOSED LOOP CEE MATRIX IS -"
           CALL PRINTM(L,NL,CEE)
C
```

VITA

Thomas Lewis was born 27 January 1958 in Albert Lea, Minnesota. He graduated from Suffield Academy, Suffield Connecticut in 1976, and then attended Purdue University, West Lafayette Indiana, where he earned a Bachelor of Science degree in Aeronautical and Astronautical Engineering, graduating May 1981. He also received a reserve commission in the USAF through the ROTC program at Purdue. He immediately pursued a graduate degree by enrolling in the Air Force Institute of Technology, Department of Aeronautical Engineering, WPAFB Ohio, in June 1981.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
AFIT/GAE/EE/82D-1

2. GOVT ACCESSION NO.
AD-A124871

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
HIGH-GAIN ERROR ACTUATED FLIGHT CONTROL SYSTEMS FOR CONTINUOUS LINEAR MULTIVARIABLE PLANTS

5. TYPE OF REPORT & PERIOD COVERED
MS Thesis

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
2Lt. Thomas Lewis, USAF

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Air Force Institute of Technology
(AFIT/EN) Wright-Patterson AFB OH 45433

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

12. REPORT DATE
December, 1982

13. NUMBER OF PAGES
137

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release: IAW AFR 190-17.
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

4 JAN [illegible]

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Flight Controller; Decoupling, non-interacting; Singular Pertubations; Advanced Fighter Aircraft, AFTI; Proportional plus Integral Control; Transmission ZEros, Decoupling Zeros

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The theory of high-gain error actuated feedback control, developed by Porter and Bradshaw, is applied to the design of various lateral-directional decoupling flight control systems for an advanced aircraft. The controllers developed in this report utilize output feedback with proportional plus integral control to produce desirable closed-loop responses with minimal interaction between outputs. Because of the structure of the system, measurement variables in addition to the outputs are necessary to apply this method.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. The report examins controller design robustness by varying the flight conditionds or maneuver commands from the ones the controller was specifically designed for, and then judges system performance. The results show that the controller is robust with respect to varying flight conditions, but is not robust with respect to varying maneuver commands. This report also examins the effect of first-order actuator dynamics in the system model. Actuator dynamics are shown to significantly affect the control system response, indicating that a simplified model, without actuators, is not desirable in ones control design scheme. Also a computer program to determine transmission zeros and decoupling zeros is developed.

END

FILMED

3-83

DTIC